



A University of Sussex DPhil thesis

Available online via Sussex Research Online:

<http://sro.sussex.ac.uk/>

This thesis is protected by copyright which belongs to the author.

This thesis cannot be reproduced or quoted extensively from without first obtaining permission in writing from the Author

The content must not be changed in any way or sold commercially in any format or medium without the formal permission of the Author

When referring to this work, full bibliographic details including the author, title, awarding institution and date of the thesis must be given

Please visit Sussex Research Online for more information and further details

Learning programming via worked-examples: The effects of cognitive load and learning styles

Siti Soraya ABDUL RAHMAN
(Student registration no: 20710654)
Human-Centred Technology Group (IDEAs Lab)
Department of Informatics
School of Engineering and Informatics
University of Sussex, UK

19th September 2011

Acknowledgements

All praise be to Allah for His mercy and blessings upon me. I am grateful to Allah the Almighty for granting me the strength and perseverance to complete this thesis and thus getting a degree of Doctor of Philosophy in Cognitive Science from the University of Sussex.

Special thanks go to my husband, Mstr. Mohd Adnin for his love and support to get me through the hard times in PhD life. Thank you for your precious time in taking care of our lovely children. To my children, Siti Umairah and Muhammad Danial, I owe you big time after all your patience and understanding. For my 'baby PhD', Mir Zara, thank you for giving me the strength and courage to accomplish this study.

My heartfelt thanks go to both my parents to whom I owe a debt of gratitude for their blessings and prayers, Haji Abdul Rahman and Madam Halina Lajium.

I am sincerely thankful to my supervisor, Emeritus Professor Benedict du Boulay who did a brilliant job of supervising and, not least because his priceless advice over the past four years of my life as a PhD student at the University of Sussex. A big thank you also goes to Professor Andy Field for his invaluable guidance through the statistics.

Notable thanks are due to the University of Malaya and the Ministry of Higher Education (Malaysia) for granting me a scholarship to pursue a Doctoral programme in the UK.

Lastly, thanks to Mr. Regan Rajan for his dedication and a magnificent job in developing LECSES. Without it, I would not have been able to conduct the experiment!

TABLE OF CONTENTS

LIST OF TABLES	VII
LIST OF FIGURES	X
ABSTRACT	XII
CHAPTER 1 INTRODUCTION	1
1.1 Statement of the problem	2
1.2 Purpose of this research.....	5
1.3 Aims and objectives of the present research	5
1.4 Methodology of the present research	6
1.5 Significance of the present research	8
1.6 The organisation of the thesis	8
CHAPTER 2 LEARNING PROGRAMMING VIA WORKED-EXAMPLES	12
2.1 Introduction	12
2.2 Analogical problem solving and transfer.....	12
2.3 Learning from worked-examples	14
2.3.1 Instructional principles from worked-example research.....	16
2.4 Cognitive load and working memory	17
2.4.1 Measuring cognitive load	20
2.4.1.1 Subjective measures.....	21
2.4.1.2 Task- and performance-based or secondary task performance	21
2.4.1.3 Physiological techniques.....	22
2.4.1.4 Performance on transfer and efficiency measure	22
2.4.1.5 Time on task.....	24
2.4.1.6 Measuring three different cognitive loads separately	24
2.5 Cognitive and learning styles: effects on programming performance	26
2.5.1 Learning styles models	26
2.5.2 Individual learner differences and programming performance.....	28
2.6 Related work	30
2.6.1 Example-based learning systems.....	30
2.6.1.1 Example-based Programming System (EBPS)	30
2.6.1.2 Example Tool (ET)	31
2.6.1.3 Episodic Learner Model Programming Environment (ELM-PE)	31
2.6.1.4 Episodic Learner Model Adaptive Remote Tutor (ELM-ART)	32
2.6.1.5 WebEx.....	33
2.6.1.6 Structural Example-based Adaptive Tutoring Systems (SEATS).....	33
2.6.1.7 CORT (Code Restructuring Tool)	33
2.6.1.8 Bridge	34
2.6.2 Cognitive load theory as a basis for the instructional design of a programming course.....	35

2.7	Conclusion.....	36
CHAPTER 3 RESEARCH QUESTIONS AND HYPOTHESES		43
3.1	Introduction	43
3.2	An exploratory pilot study	44
3.2.1	Method	44
3.2.1.1	Learning style inventory	44
3.2.1.2	Participants	45
3.2.1.3	Procedure	47
3.2.1.4	Coding observational data	49
3.2.2	Results.....	51
3.2.2.1	Correlation analysis	52
3.2.2.2	Programming tasks performance	53
3.2.2.3	Observation of the participants	55
3.3	Planning for the main experiment	56
3.4	Research questions and hypotheses.....	61
3.4.1	The H1 hypothesis	62
3.4.2	The H2 hypothesis	62
3.4.3	The H3 hypothesis	63
3.4.4	The H4 hypothesis	63
3.4.5	The H5 hypothesis	64
3.5	Conclusion	66
CHAPTER 4 LECSES		67
4.1	Introduction	67
4.2	The paired-method strategy.....	67
4.2.1	Theoretical assumptions and rationale for the design of the Paired-method strategy	70
4.2.2	Related work underlying the interface design	73
4.3	Web-based worked-example system: LECSES.....	75
4.3.1	The development environment.....	75
4.3.2	The interface	78
4.3.2.1	Structure-emphasising strategy.....	79
4.3.2.2	Completion strategy.....	88
4.3.3	The editor	95
4.3.4	The report generator.....	98
4.3.5	The administrator module	99
4.4	Conclusion	100
CHAPTER 5 EXPERIMENTAL DESIGN.....		101
5.1	Introduction	101
5.2	Pilot experiment	102
5.3	The design of the main experiment	105
5.3.1	Phases and procedure of the main experiment	105

5.3.1.1	Pre-experimental phase	106
5.3.1.2	Learning phase	107
5.3.1.3	Transfer phase	108
5.3.1.4	Post-experimental phase	108
5.3.2	LECSSES learning environment, experimental materials and variations	109
5.3.2.1	Experimental materials	109
5.3.2.2	Experimental variations	114
5.3.3	Instruments.....	119
5.3.3.1	Index of Learning Styles inventory (ILS)	119
5.3.3.2	Operation word span (OSPAN).....	120
5.3.3.3	Programming pre-test	121
5.3.3.4	The 5-point rating scale for cognitive load measures	121
5.3.4	Participants.....	122
5.3.5	Scoring	122
5.4	Proposed statistical analyses and measured variables	123
5.4.1	Web-OSPAN	123
5.4.2	Learning phase	125
5.4.3	Transfer phase	127
5.4.4	Learning <i>outcome</i> efficiency	130
5.4.5	Learning <i>process</i> efficiency	130
5.4.6	Task involvement	131
5.5	Conclusion	132
CHAPTER 6 THE LEARNING PROCESS		133
6.1	Introduction.....	133
6.2	Overview of research questions and hypotheses for the learning phase	134
6.3	Methods.....	136
6.3.1	Participants.....	136
6.3.2	Materials	138
6.3.2.1	Assessment of Learning Style.....	138
6.3.2.2	Assessment of working memory capacity	138
6.3.2.3	Assessment of cognitive load.....	138
6.3.2.4	Programming pre-test	139
6.3.2.5	Worked-example problems	139
6.3.3	Method for statistical data analysis	140
6.3.3.1	Analysis for Web-OSPAN measures.....	142
6.3.3.2	Analysis for comparing the effects of the strategies and learning styles	143
6.4	Effort and difficulty scores.....	144
6.5	Results.....	149
6.5.1	Internal validity	151
6.5.2	Measures from Web-OSPAN	152

6.5.3	Comparing the effects of the strategies	155
6.5.3.1	The analysis of results of total effort/difficulty scores.....	156
6.5.3.2	The analysis of results of overall effort/difficulty scores.....	159
6.5.3.3	The analysis of results for repeated measures for effort/difficulty scores.....	162
6.5.4	Comparing the effects of the learning styles	163
6.5.4.1	The alternative (H4-A1) and (H4-A2) hypotheses.....	164
6.5.4.2	The null (H4-01) hypothesis	169
6.5.4.3	The null (H4-02) hypothesis	171
6.5.4.4	The null (H4-03) hypothesis	175
6.5.4.5	The post-hoc comparison.....	177
6.6	Conclusion	182
CHAPTER 7	LEARNING <i>OUTCOME</i>	183
7.1	Introduction	183
7.2	Overview of the research hypotheses for the transfer phase	183
7.3	Mental effort scores, time on tests, and post-test scores	184
7.4	Results.....	190
7.4.1	Internal validity	191
7.4.2	Comparing the effects of the strategies	191
7.4.2.1	The analysis of the correlation between ILS values and reported mental effort	193
7.4.2.2	The analysis of the correlation between ILS values and post-test scores (and time on tests)	193
7.4.2.3	The analysis of the correlation between time on tests and post-test scores.....	195
7.4.2.4	The analysis of results investigating any differences across the strategy groups.....	196
7.4.2.5	The analysis of results for repeated measures for mental effort scores and post-test scores.....	197
7.4.3	Comparing the effects of the learning styles	199
7.4.3.1	The null (H5-01) hypothesis	200
7.4.3.2	The null (H5-02) hypothesis	202
7.4.3.3	The null (H5-03) hypothesis	205
7.4.3.4	The alternative (H5-A1) hypothesis.....	207
7.4.3.5	The alternative (H5-A2) hypothesis.....	210
7.4.4	Instructional efficiency measures	212
7.4.4.1	Learning process efficiency	213
7.4.4.2	Task involvement.....	213
7.4.4.3	Learning outcome efficiency	213
7.4.4.4	Efficiency measures for active learners in the three strategy groups	215
7.4.4.5	Efficiency measures for reflective learners in the three strategy groups.....	217
7.4.5	Comparing effects of prior knowledge.....	218
7.5	Conclusion	222
CHAPTER 8	PUTTING THE EXPERIMENTAL RESULTS IN CONTEXT.....	226
8.1	Introduction	226
8.2	Current formulation of cognitive load theory.....	226

8.3	Self-reporting on cognitive load measures	227
8.4	Individual differences: Learning styles and working memory capacity	231
8.5	Comparing the effects of the strategies	234
8.5.1	Completion strategy	236
8.5.2	Paired-method strategy	237
8.5.3	Structure-emphasising strategy	239
8.6	Comparing the effects of the learning styles	242
8.7	Cognitive load effects	244
8.8	Conclusion	250
CHAPTER 9	CONCLUSIONS.....	254
9.1	Contributions of the thesis	254
9.2	Research limitations	256
9.2.1	Sample size	256
9.2.2	WMC and ILS instruments	257
9.2.3	LECSSES	258
9.2.4	Other limitations	260
9.3	Future work	261
APPENDIX A	PARTICIPANT CONSENT FORM	263
APPENDIX B	QUESTIONNAIRE ON PROGRAMMING BACKGROUND	264
APPENDIX C	INDEX OF LEARNING STYLES INSTRUMENT (ENGLISH VERSION)	265
APPENDIX D	INDEX OF LEARNING STYLES INSTRUMENT (MALAY VERSION)	270
APPENDIX E	PAPER-BASED INSTRUMENT FOR RECORDING WEB-OSPAN SCORES	274
APPENDIX F	PROGRAMMING PRE-TEST INSTRUMENT	275
APPENDIX G	EXAMPLE PROBLEMS FOR THE LEARNING PHASE	279
APPENDIX H	TRANSFER PROBLEMS FOR THE TRANSFER PHASE	312
APPENDIX I	QUESTIONNAIRE ON WORKED-EXAMPLE STRATEGIES AND LECSSES	321
APPENDIX J	LIST OF PUBLICATIONS	325
REFERENCES	326

LIST OF TABLES

Table 2.1 : A review of the worked-example design and strategy	37
Table 3.1 : Frequency for the ILS scores	45
Table 3.2 : Preferences for learning styles on the Active/Reflective dimension of the ILS	46
Table 3.3 : 4-point rating scale for observational data analysis	49
Table 3.4 : Coding categories and scheme for analysing observational data	50
Table 3.5 : Factors assessed (that influence the use of example programs) in the pilot study	51
Table 3.6 : Results of the correlations between learning style scores and the factors	52
Table 3.7 : Marking scheme for Task 2	54
Table 3.8 : Results of programming task performance (active learners)	54
Table 3.9 : Results of programming task performance (reflective learners)	54
Table 3.10: Research hypotheses	65
Table 4.1 : A summary of the design of the Paired-method strategy	68
Table 4.2 : Feedback comments	95
Table 4.3 : Administrator module for LECSES	100
Table 5.1 : The quality of explanations and total correct answers	103
Table 5.2 : Average solving time and reported effort / difficulty (<i>rounded</i>)	104
Table 5.3 : Worked-example problems for the learning phase	112
Table 5.4 : Word problems statement for the transfer phase	118
Table 5.5 : Marking scheme for Bus Ticket Machine problem	122
Table 6.1 : Participants' programming backgrounds	137
Table 6.2 : Methods for comparison analysis	144
Table 6.3 : Tests of normality (Learning phase)	146
Table 6.4 : Tests of homogeneity of variance (Learning phase)	147
Table 6.5 : Hypotheses and planned comparisons	149
Table 6.6 : Descriptive statistics for pre-test scores for the three strategy groups	151
Table 6.7 : Descriptive statistics for Web-OSPAN measures for the three learning styles	153
Table 6.8 : Results from Spearman's rho investigating correlation between WMC values and other measures of Web-OSPAN	153
Table 6.9 : Results from Spearman's rho investigating correlation between ILS values and Web-OSPAN measures	154
Table 6.10: Descriptive statistics for effort and difficulty scores (<i>total</i> and <i>overall</i>) for the three strategy groups	155
Table 6.11: Results from Spearman's rho correlation and Mann-Whitney U test investigating <i>total</i> difficulty scores (Set 2)	156
Table 6.12: Results from Kruskal-Wallis and Mann-Whitney U test investigating <i>total</i> difficulty scores (Set 3)	157
Table 6.13: Results from Kruskal-Wallis and Mann-Whitney U test investigating <i>total</i> effort scores (Set 3) ..	158
Table 6.14: Results from Kruskal-Wallis and Mann-Whitney U test investigating <i>overall</i> difficulty scores ...	159

Table 6.15: Results from Wilcoxon Signed Ranked test investigating change in scores on the effort/difficulty scale across two sets of worked-example problem (i.e. Set 2 and 3)	162
Table 6.16: Descriptive statistics for effort and difficulty scores (<i>total/overall</i>) for active and reflective in the three strategy groups	163
Table 6.17a: Results from Mann-Whitney U test investigating <i>H4-A1</i>	164
Table 6.17b: Results from Mann-Whitney U test investigating <i>H4-A2</i>	165
Table 6.18: Results from Mann-Whitney U test investigating <i>H4-01</i>	170
Table 6.19: Results from Mann-Whitney U test investigating <i>H4-02</i>	171
Table 6.20: Results from Mann-Whitney U test investigating <i>H4-03</i>	175
Table 6.21: Results from Mann-Whitney U test of <i>post-hoc</i> comparisons	177
Table 6.22: A summary of analysis for comparing the effects of the strategies	179
Table 6.23: A summary of analysis for comparing the effects of the learning styles	180
Table 7.1 : Tests of normality (Transfer phase)	187
Table 7.2 : Tests of homogeneity of variance (Transfer phase)	187
Table 7.3 : Hypotheses and planned comparisons	190
Table 7.4 : Descriptive statistics for mental effort, time on tests, and post-test scores (near and far transfer) for the three strategy groups	192
Table 7.5 : Results from Spearman's rho correlation and Mann-Whitney U test (Transfer phase)	192
Table 7.6 : Results from Wilcoxon Signed Ranked test investigating change on the <i>total</i> mental effort scores and <i>total</i> post-test scores across the two sets of transfer tests (i.e. near and far transfer)	197
Table 7.7 : Descriptive statistics for mental effort, time on tests, and post-test scores (near and far transfer) for active and reflective in the three strategy groups	199
Table 7.8 : Results from Mann-Whitney U test investigating <i>H5-01</i>	200
Table 7.9 : Mean rank and median for Active and Reflective learners in Group P	200
Table 7.10: Results from Mann-Whitney U test investigating <i>H5-02</i>	202
Table 7.11: Mean rank and median for Reflective learners in Group S and P	203
Table 7.12: Results from a Mann-Whitney U test investigating <i>H5-03</i>	205
Table 7.13: Mean rank and median for Reflective learners in Group C and P	205
Table 7.14: Results from Mann-Whitney U test investigating <i>H5-A1</i>	207
Table 7.15: Mean rank and median for Active and Reflective learners in Group S	208
Table 7.16: Results from Mann-Whitney U test investigating <i>H5-A2</i>	210
Table 7.17: Mean rank and median for Active and Reflective learners in Group C	210
Table 7.18: Descriptive statistics for reported difficulty/effort/mental effort, performance, and efficiency means for the three strategy groups	212
Table 7.19: Descriptive statistics for reported difficulty/effort/mental effort, performance, and efficiency means for active learners in the three strategy groups	215
Table 7.20: Descriptive statistics for reported difficulty/effort/mental effort, performance, and efficiency means for reflective learners in the three strategy groups	217
Table 7.21: Descriptive statistics for pre-test scores for low and high prior knowledge (post-hoc categorised) in the three strategy groups	218

Table 7.22: Results from Spearman's rho investigating correlation between pre-test scores and <i>overall</i> post-test scores for the three strategy groups	219
Table 7.23: Results from the Mann-Whitney U test investigating <i>overall</i> post-test scores of low and high prior knowledge learners	220
Table 7.24: Results from T-test investigating <i>overall</i> post-test scores on low and high prior knowledge learners.....	220
Table 7.25: A summary of analysis for comparing the effects of the strategies	224
Table 7.26: A summary of analysis for comparing the effects of the learning styles	224

LIST OF FIGURES

Figure 1.1 : Research methodology process	7
Figure 2.1 : The active/reflective dimension of the ILS	27
Figure 3.1 : Distribution of preferences for Active (left end) / Reflective (right end) on the ILS dimension	46
Figure 3.2 : The programming tasks	48
Figure 3.3 : The difficulties inherent in learning via example programs	57
Figure 4.1 : Client server architecture	76
Figure 4.2 : The learning interaction (the Paired-method strategy)	79
Figure 4.3 : An example problem consisting of problem description, a sample run, and an example solution (explanation exercise)	80
Figure 4.4 : Invisible plan structures via masking mechanism	81
Figure 4.5 : A prompt dialog box for an explanation input	82
Figure 4.6 : Hint(s) on demand	83
Figure 4.7 : Editing explanation	84
Figure 4.8 : Predict the program's behaviour (reflection exercise)	85
Figure 4.9 : Reflection exercise	86
Figure 4.10: A questionnaire page	87
Figure 4.11: An example problem consisting of problem description, a sample run, and an example solution (completion exercise)	89
Figure 4.12: Text box in completion exercise	90
Figure 4.13: Feedback on the correctness of answer(s)	91
Figure 4.14: Lines of code that are initially invisibly sensitive to being clicked on	92
Figure 4.15: Feedback comments dialog box	94
Figure 4.16: Editor for composing the explanation/reflection exercises	95
Figure 4.17: Composing an example problem	96
Figure 4.18: Editor for composing a completion exercise	97
Figure 4.19: Editor for composing a modification exercise	98
Figure 4.20: LECSES Report generation	99
Figure 5.1 : Phases and procedure of the experiment	106
Figure 5.2 : Experimental materials	110
Figure 5.3 : Different tactics of prompting questions for the counter loop plan	111
Figure 5.4 : Excerpt from a program solution for reflection exercise	115
Figure 5.5 : The active/reflective dimension of the ILS	119
Figure 5.6 : Exploring relationships between WMC values and other Web-OSPAN measures	124
Figure 5.7 : Exploring relationships between ILS values and Web-OSPAN measures	124
Figure 5.8 : Exploring relationships between ILS category and WMC category	124
Figure 5.9 : Exploring relationships between ILS values and effort (and difficulty) variables	125
Figure 5.10: Exploring differences between strategy groups on effort and difficulty variables	125
Figure 5.11: Exploring repeated measures for effort and difficulty variables	126

Figure 5.12: Exploring between-within differences on effort and difficulty variables	126
Figure 5.13: Exploring relationships between ILS values and mental effort/time on tests/post-tests	127
Figure 5.14: Exploring relationships between post-tests and mental effort/time on tests	127
Figure 5.15: Exploring differences between strategy groups on mental effort/time on tests/post-tests	128
Figure 5.16: Exploring repeated measures for mental effort and post-tests variables	128
Figure 5.17: Exploring between-within differences on mental effort/time on tests/post-tests	129
Figure 5.18: Exploring relationships between pre-test and post-test variables	129
Figure 5.19: Exploring between and within differences on overall post-tests	129
Figure 6.1 : Experimental materials	140
Figure 6.2 : The active/reflective dimension of the ILS	141
Figure 6.3 : Variables for the learning phase	145
Figure 6.4 : Overall effort scores - Group S and C, distinguishing between the active and reflective learners	167
Figure 6.5 : Overall difficulty scores - Group S and C, distinguishing between the active and reflective learners	168
Figure 6.6 : Overall effort scores (<i>Mdn</i>) of reflective learners in Groups C and P	173
Figure 6.7 : Overall difficulty scores (<i>Mdn</i>) of reflective learners in Groups C and P	174
Figure 7.1 : Variables for the transfer phase	185
Figure 7.2 : Learning process and <i>outcome</i> efficiencies for the three strategy groups	214
Figure 7.3 : Task involvement for the three strategy groups	214
Figure 7.4 : Learning process and <i>outcome</i> efficiencies for the active learners	216
Figure 7.5 : Task involvement for the active learners	216
Figure 7.6 : Learning process and <i>outcome</i> efficiencies for the reflective learners	217
Figure 7.7 : Task involvement for the reflective learners	218
Figure 7.8 : Interaction effect in the Completion strategy	221

Abstract

This research explored strategies for learning programming via worked-examples that promote schema acquisition and transfer. However, learning style is a factor in how much learners are willing to expend *serious effort* on understanding worked-examples, with active learners tending to be more impatient of them than reflective learners. It was hypothesised that these two learning styles might also interact with learners' cognitive load. The research proposed a worked-example format, called a Paired-method strategy that combines a Structure-emphasising strategy with a Completion strategy. An experiment was conducted to compare the effects of the three worked-examples strategies on cognitive load measures and on learning performance. The experiment also examined the degree to which individual learning style influenced the learning process and performance. Overall, the results of the experiment were inconsistent. In comparing the effects of the three strategies, there were significant differences in reported difficulty and effort during the learning phase, with difficulty but not effort in favour of the Completion strategy. However no significant differences were detected in reported mental effort during the post-tests in the transfer phase. This was also the case for the performance on the post-tests. Concerning efficiency measures, the results revealed significant differences between the three strategy groups in terms of the learning *process* and task involvement, with the learning *process* in favour of the Completion strategy. Unexpectedly, no significant differences were observed in learning *outcome* efficiencies. Despite this, there was a trend in the data that suggested a *partial reversal* effect for the Completion strategy. Moreover, the results partially replicated earlier findings on the explanation effect. In comparing the effects of the two learning styles, there were no significant differences between active and reflective learners in the three strategy groups on cognitive load measures and on learning performance (nor between reflective learners in the Paired-method strategy and the other strategies). Finally, concerning efficiency measures, there was a significant difference between active learners in the three strategy groups on task involvement. Despite all these, effect sizes ranging from a medium to large suggested that learning styles might have interacted with learners' cognitive load.

Chapter 1 Introduction

Many learners find learning programming difficult, not least because of the “abstract nature of the programming task” (Dunican, 2002 p. 89). Abstract concepts such as variables, data types, and dynamic memory are not closely related to real world situations and so learners find these concepts difficult to grasp (Dunican, 2002). According to Perkins and Martin (1986), as reported by Gilmore (1990), the main difficulties are two-fold: *fragile knowledge* and *neglected strategies*, to use Perkins and Martin’s (1986) phrases. The former is “knowledge that the student has, but fails to use when it is needed”. In the latter case, Carbone, Hurst, Mitchell and Gunstone (2001) found that insufficient strategies result in learners failing in their attempts at solving a programming task. Carbone, Hurst et al., (2001) further argued that this can occur at three different points: the initial stage of designing a solution, during coding of the solution, and finally at debugging run time errors.

These are just some of the problems faced by learners that affect their performance in an introductory programming course. Indeed, much research has been conducted to study the difficulties learners have in learning to program and/or factors that influence programming performance (e.g. Bennedsen & Caspersen, 2006; Lahtinen, Ala-Mutka & Järvinen, 2005; Bergin & Reilly, 2005; Mancy & Reid, 2004). Among many others is the research done in the area of analogical problem solving.

1.1 Statement of the problem

The use of examples is one of three types of analogical reasoning in problem solving (Mayer, 1992; Reimann & Schult, 1996). Past research has shown that examples play an important role in learning and problem solving (e.g. Pirolli & Anderson, 1985; Chi, Bassok, Lewis, Reimann & Glaser, 1989) and are crucial to the acquisition of initial cognitive skills (Atkinson, Derry, Renkl & Wortham, 2000) and hence play an important role in the learning of programming. The construction of schemata is one of the underlying processes in acquiring and managing such skills (van Merriënboer & Paas, 1990).

Learning via worked-examples has received a significant amount of interest from researchers, including researchers in the programming education domain and indeed some have developed web-based systems to support such learning (e.g. Weber, 1993; Chang, Chiao, Chen & Hsiao, 2000; Weber & Brusilovsky, 2001; Brusilovsky, 2001; Davidovic, Warrant & Trichina, 2003; Garner, 2007). Nevertheless, evidence from worked-example research indicates several limitations of example-based learning. Although several systems have attempted to address these limitations, various questions remain open. For instance, it is not clear whether these systems, with the exception of the work done by Chang, Chiao et al. (2000) and Davidovic, Warren et al. (2003) have been sufficiently evaluated against learning outcomes (in terms of the quality of acquired cognitive schemata, including transfer). More importantly, the relationship between individual learning styles and learning outcomes (and cognitive load effects) resulting from using the system needs elucidation. Note that Graf, Lin and Kinshuk (2008) have identified an *indirect relationship* between working memory capacity and learning styles, as drawn from the literature. That is to say, learners with low working memory capacity tend to prefer an active style of learning, on the contrary, learners with high

working memory capacity tend to prefer a reflective style of learning. Note also that an instructional design based on cognitive load theory (CLT) argues for the careful utilisation of working memory capacity to encourage more effective schema construction (Sweller, van Merriënboer & Paas, 1998).

In addition, previous evidence in the area of learning styles research has suggested that learners differ in the ways they perceive and process information, as well as respond to and interact with their learning environment (e.g. Keefe, 1979, as quoted by Felder and Spurlin, 2005). In line with this, several empirical findings within the programming education literature have pointed out that reflective learners perform better than active learners in an introductory computer science course and/or in programming performance (e.g. Allert, 2004; Thomas, Ratcliffe, Woodbury & Jarman, 2002; Chamillard & Karolick, 1999; van Merriënboer, 1988). One noteworthy point concerning learning styles is the lack of a clear relationship between reflection/impulsivity (i.e. reflective/active) and teaching methods. Thus, the question arises as to whether learners should be given what they like as in the use of teaching methods that provide an opportunity for impulsive learners to be active, or whether learners should be given what should be helpful for them, as in the use of teaching methods that force impulsive learners to behave more reflectively (van Merriënboer, 2009)¹. Hence, we argued that investigating the relationship between learning styles and working memory capacity, and so the differential effects of cognitive load on active and reflective learners, appeared to be a promising for research, and it became the main focus of this thesis.

¹ Personal communication, 14/11/2009

Another issue worth exploring is why examples are so seldom used by learners (see Weber & Brusilovsky, 2001) and are often neglected in programming instruction (van Merriënboer & Paas, 1990) given the fact that these are an effective way to learn a complex cognitive skill such as problem solving (Paas & van Gog, 2006). As a final point, only a limited amount of research on instructional design involving worked-examples has been carried out in the area of programming education (e.g. van Merriënboer, 1990b; van Merriënboer & Paas, 1990; van Merriënboer & de Croock, 1992; Trafton & Reiser, 1993).

In an attempt to improve the effectiveness of worked-examples, Atkinson, Derry et al. (2000) proposed three moderating factors. These include intra-example features, inter-example features, and individual differences in example processing (e.g. Chi, Bassok et al., 1989; Renkl, 1997). In addition to this work focusing on the instructional principles of worked-examples, recent research is also focusing on techniques to optimise the cognitive load for learning from worked-examples, (see Paas & van Gog, 2006; Moreno, 2006). Within the broader programming education literature, some studies have used cognitive load theory as a basis for teaching a programming course (e.g. Caspersen & Bennedsen, 2007; Gray, St. Clair, James & Mead, 2007) Then again, it is not clear whether the proposed instructional mechanism has been sufficiently evaluated against learning outcomes including transfer using valid and reliable measures according to cognitive load theory. Finally, recent effort has been made with respect to a model of measuring cognitive load during programming instruction and embedded in an intelligent tutoring system (e.g. Yousoof, Sapiyan & Kamaluddin, 2007).

1.2 Purpose of this research

Taking all these aspects into account, the purpose of this research is to bridge the gaps identified above by extending previous research on example-based learning systems with regard to the instructional design of the worked-examples themselves. This can be done by taking into consideration instructional principles from worked-example research (Atkinson, Derry et al., 2000) and more specifically, by drawing from assumptions laid down within the current developments of cognitive load theory. Indeed, cognitive load theory has provided guidelines for the development of several instructional formats, including worked-examples (Kirschner, 2002).

1.3 Aims and objectives of the present research

The present research sets out to explore strategies for learning programming via worked-examples, which seek to promote schema acquisition and transfer. However, we argued that learning style is a factor that influences how much *serious effort* learners are willing to expend in understanding worked-examples, with active learners tending to be more impatient than reflective learners. As a result, active learners may become overwhelmed and experience cognitive overload. In view of that, we hypothesised that the two learning styles might interact with learners' cognitive load and would determine the quality of acquired cognitive schemata and hence the transfer of learning. The hypothesis is consistent with the *indirect relationship* proposed by Graf, Lin et al., (2008) as discussed in Section 1.1. To answer this question, we investigated the differential effects of different worked-example strategies on the learning process and outcomes (including transfer) as well as on the cognitive load that occurs during learning, taking into account learners' learning styles. In so doing, the present research also investigated the relationship between the active and reflective learning styles

and learners' working memory capacity. Summing up, the main aim of the present research was to extend prior work on the design of worked-examples in the area of programming education. Specifically, it investigated the effective design of a worked-example strategy that particularly aimed at helping active learners benefit from being exposed to worked-examples thus improve their learning, and so equalise the learning outcomes of both active and reflective learners. For the purpose of the investigation, a web-based worked-example system for learning programming was built and evaluated based on the design of the worked-example strategy.

To achieve the stated aims, we sought to address the following objectives. The first objective was to investigate any differential effects on the learning process and outcomes (in terms of the quality of acquired cognitive schemata, including transfer) of different worked-example strategies, taking into account learners' learning styles. The second objective was to investigate any consequential variations of cognitive load that occur during learning by means of the valid and reliable measures derived from cognitive load theory.

1.4 Methodology of the present research

Figure 1.1 outlines the methodology used in pursuit of the aims and objectives of the present research. Note that the deliverables of each step of the research methodology process are presented in italics. The work on the thesis was started in October 2007. Phase 1 started with a literature review, followed by an exploratory pilot study, which was conducted in August 2008. The main experiment was designed and conceived by May 2009, together with the research questions and hypotheses. The principles underlying the design of the proposed worked-example strategy, named the Paired-method was finalised during the later months of

the phase. Phase 1 took 19 months (including 3 months of intermission) to complete. Phase 2 started with the design of the Paired-method strategy, followed by paper prototyping, which was piloted in January 2010. The LECSES system development was broadly initiated in February 2010 and completed by the end of July 2010. The preparations of the experimental materials were also carried out during the whole period of the phase. Phase 3 started in the middle of July 2010 when the pilot study was conducted, and this led to improvements in the design of the main experiment and its experimental materials. The phase continued with the main experiment and was completed by the end of August 2010. Phase 4 started in September 2010 and completed by middle of September 2011 (including the thesis write up throughout the period of the phase).

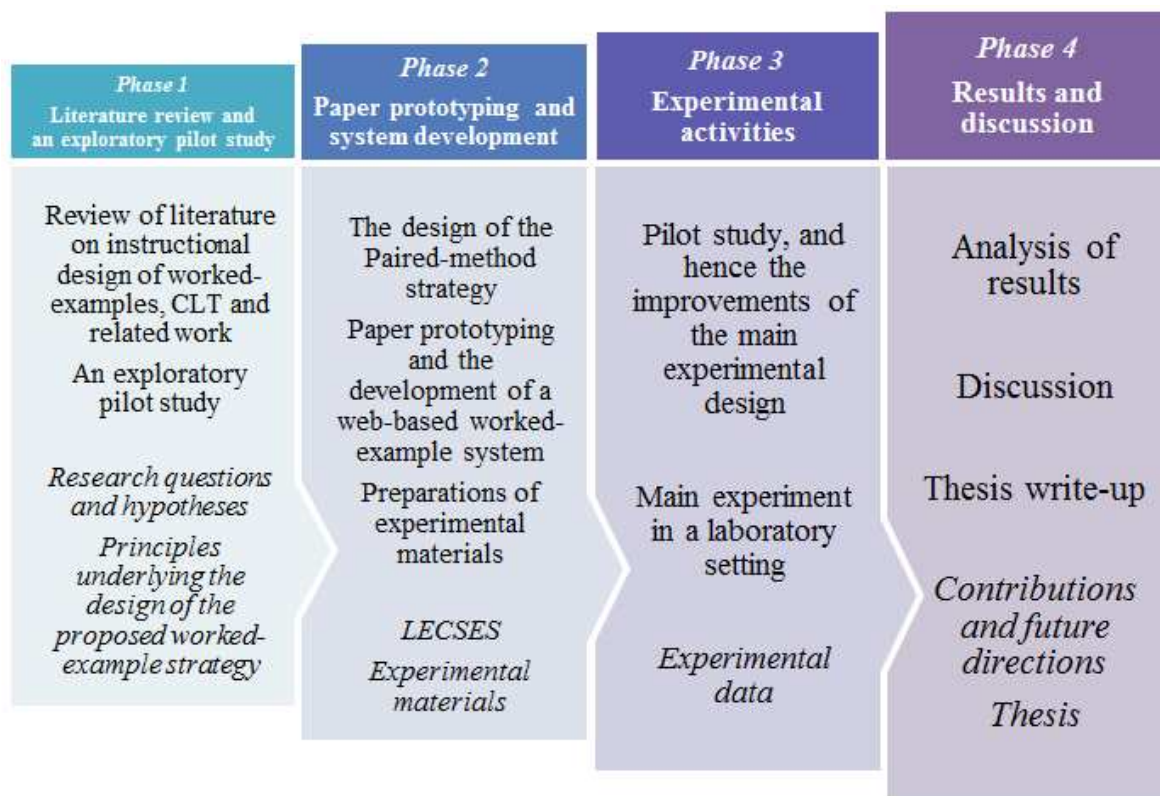


Figure 1.1: Research methodology process

1.5 Significance of the present research

The effectiveness of the instructional design of worked-examples and strategies has been widely studied within the framework of cognitive load theory. It is worth noting, however, that the majority of these studies have failed to provide consistent findings with regard to the learning process and/or outcomes as well as cognitive load effects (Paas & van Gog, 2006; Moreno, 2006). See Moreno (2006) for an extensive review of this issue. The research described here advances our knowledge about cognitive load theory and provides more viable and alternative explanations for interpreting previous research findings. That is, the research help to provide a better understanding of the different kinds of cognitive load that occur during learning with different worked-example strategies for the active and reflective learners. Besides, the research provides both theoretical and practical implications for learning via worked-examples in the area of programming instruction and learning styles. Also, it provides preliminary work towards a macro-adaptive system to support such learning.

1.6 The organisation of the thesis

Cognitive load theory developed substantially over the period during which when the thesis was undertaken. The literature review presented in Chapter 2 covers research up to the year 2009, as this influenced the design of the main experiment. Note that, the main experiment was designed and conceived in the year 2009. Note also that the first few sections of Chapter 4 cover the literature underpinning the detailed design of the worked-example strategy. The analyses and discussion of the main experiment was carried out in September 2010 when the new cognitive load theory formulation (Sweller, 2010) and other related work (e.g. de Jong, 2010; Moreno, 2010; van Gog & Rummel, 2010) emerged. So Chapters 8 and 9 cover recent

research published from the year 2010. Research published before the year 2010 is also included wherever relevant.

Chapter 1 has provided a general overview of the present research, covering a statement of the problem, and the aims and objectives of the research. The chapter also highlights the significance of the research as well as its contribution in various fields.

Chapter 2 provides a review of the literature which covers interdisciplinary areas of research. More specifically, it covers research in the area of analogical problem solving and transfer, from worked-example research to cognitive load theory as well as learning styles, among others. Then, the chapter presents related work within the domain of programming education. Next, the chapter reviews worked-example design and strategies within the context of the instructional principles derived from worked-example research and/or cognitive load theory-inspired research.

Chapter 3 presents an observational pilot study that explored the context in which learners make use of examples as they solve programming problems. In particular, the aim was to understand learners' behaviour and the consequences of this behaviour on their problem solving. The findings from the pilot study guided the development of the research questions and hypotheses. The chapter also discusses problems associated with learning from worked-examples as derived from the pilot study as well as from the literature. Finally, suggestions are made with regard to the design of the proposed worked-example strategy.

Chapter 4 starts with a discussion of the theoretical background for the design of worked-example strategies, followed by related work underlying the interface design of a web-based worked-example system. Next, the chapter presents the web-based worked-example system, LECSES (Learning Examples using Completion and Structure-emphasising Strategies). The LECSES development environment is described first. Then, the two types of interface supporting each of the strategies are discussed, with screenshots that illustrate the learning interaction. Finally, the LECSES editor, report generator, and its main modules are presented.

Chapter 5 starts by briefly describing a second pilot experiment, which led to improvements in the design of the main experiment and its experimental materials. Next, the chapter discusses the main experimental design, covering its phases (learning and transfer), procedures, the experimental materials, instruments, and its participants. The subsequent sections briefly explain the proposed statistical analyses and dependent variables.

Chapter 6 starts with a synopsis of the research questions and hypotheses for the learning phase. Then, the chapter presents a detailed description of the experimental methods and statistical analysis techniques for the phase. Next, the chapter discusses the experimental results of the learning phase, comparing the effects of the strategies with or of the learning styles. The chapter also discusses the analysis of the results of working memory capacity measures.

Chapter 7 starts with a synopsis of research questions and hypotheses for the transfer phase. Next, the chapter presents a detailed description of the experimental methods and statistical analysis techniques for the phase. Then, the chapter discusses the experimental results of the

transfer phase, comparing the effects of the strategies with or of the learning styles, instructional efficiency measures, and differential effects of learners' prior knowledge.

Chapter 8 reflects on the presents study. It presents the experimental results in context. More specifically, the chapter discusses the research findings of the thesis in the light of recent work, particularly de Jong's (2010) criticisms.

Chapter 9 presents the research contributions of the thesis. It considers the research limitations and finally concludes with potential research questions and future directions.

Finally, the thesis provides list of references cited as well as appendices as follows:

Appendix A Participant consent form.

Appendix B Questionnaire on programming background.

Appendix C Index of Learning Styles instrument (Felder & Soloman, n.d.).

Appendix D Index of Learning Styles instrument – Malay version.

Appendix E Paper-based instrument for recording Web-OSPAN scores.

Appendix F Programming pre-test instrument.

Appendix G Example problems for the learning phase.

Appendix H Transfer problems for the transfer phase.

Appendix I Questionnaire on worked-example strategies and LECSES.

Appendix J List of publications.

Chapter 2 Learning programming via worked-examples

2.1 Introduction

Chapter 2 provides a review of the literature which covers interdisciplinary areas of research. More specifically, this chapter builds on the research in the area of analogical problem solving and transfer, from worked-example research to cognitive load theory as well as learning styles, among others. The chapter also presents related work within the domain of programming education, covering example-based learning systems and work done in the area of instructional design of a programming course. A review of the worked-example design and strategy summarised within the context of instructional principles from the worked-example research and/or cognitive load theory-inspired research conducted in the past is given at the end of this chapter. This review is largely based on Atkinson, Derry et al. (2000)'s article and related studies from literature.

2.2 Analogical problem solving and transfer

Similar problems solved in the past are often used to guide and solve the current problem (Ross, 1989a; Novick, 1988). For example, Neal (1989) observed that programmers do programming by referring to program fragments in books and manuals, and every so often they reuse or revise the code that was written previously. Among others, Pirolli and Anderson (1985) identified the role of analogical problem solving to worked-examples in the attempt by novice programmers to write recursive functions.

In fact, novices, most of the time and regardless of subject area use problem solving by analogy extensively (Reimann & Schult, 1996). Moreover it is a preferred mode of learning by novices (Renkl, Atkinson & Maier, 2000; Renkl & Atkinson, 2002).

The use of examples is one of three types of analogical reasoning in problem solving (Mayer 1992; Reimann & Schult, 1996). According to Atkinson, Derry et al. (2000) learning from examples is crucial to the acquisition of initial cognitive skills and this can be further explained by referring to a four-stage model derived from a framework called ACT-R (Anderson, Fincham & Douglass, 1997). Atkinson, Derry et al. (2000) further state that, according to this model, learning from examples is so important compared to standard problem solving that the first two stages are crucial when a learner is in the initial stage of skill acquisition. In the first stage, learners solve the problem analogically, that is to say, learners refer to some known examples and attempt to relate them to the problem at hand. In the second stage, learners develop abstract declarative rules or schemas, which may guide them in subsequent problem solving (as cited in Atkinson, Derry et al., 2000).

“A schema can be conceptualised as a cognitive structure that enables problem solvers to recognise problems as belonging to a particular category of problems that require particular operations to reach a solution. Acquired schemata can provide analogies in new problem-solving situations and can be used in mapping processes to reach solutions for unfamiliar aspects of the problem- solving task” (Paas, 1992 p. 429).

Several researchers have proposed different means to promote schema acquisition. For instance, van Merriënboer and Paas (1990) claimed that successful schema acquisition requires an investment of effort from learners and this can be achieved by providing learners with partial worked-example solutions that have to be completed. Similarly, Quilici and Mayer (2002) reported that requiring learners to abstract the underlying structural features of example problems and so organize them into a generalised problem model facilitates the learners' development of problem schemas.

Schema acquisition is regarded as one of the underlying processes in acquiring skills in programming (van Merriënboer & Paas, 1990). Soloway and Ehrlich (1984) argued that programming plans ought to be considered as schemas. Plans are generic program fragments that represent “stereotypic actions in a program” (Ehrlich & Soloway, 1984 p. 115). A plan is a chunk of programming knowledge which can be retrieved and applied in future problem solving (Rist, 1989).

2.3 Learning from worked-examples

“Worked examples are effective instructional means to teach complex problem-solving skills” (Paas & van Gog, 2006). Worked-examples or *solved example problems*, to use Schworm and Renkl's (2006) phrase are problems with a complete solution (Schworm & Renkl, 2006; Sweller, van Merriënboer et al., 1998) and according to Renkl, Stark, Gruber and Mandl (1998) consist of an example problem, some solution steps along with a final solution to the problem. Support for the idea that studying worked-examples is superior to standard problem solving has been found in numerous studies, and within the cognitive load theory literature, this is often referred to as the *worked example effect* (Sweller, van Merriënboer et al., 1998).

The worked example effect is the result of “a practice method that makes a more efficient use of learners’ limited cognitive resources than the one resulting from problem-solving practice” (Moreno, 2006 p. 171).

The benefits of worked-examples have been proposed by two different schools of theory, as cited in Trafton and Reiser (1993). That is, according to the example generalisation model, learning from examples is of importance in acquiring problem solving rules, as a result of a self-explanation process (VanLehn, Jones & Chi, 1992), and indeed more effective than unguided problem solving (Sweller & Cooper, 1985). On the other hand, the knowledge compilation model proposes that problem solving rules are created through an analogical use of the examples during problem solving (Anderson, 1987; Pirolli, 1991), and according to VanLehn, Jones et al. (1992), the learner’s explanation can then be applied in future problem solving, hence leading to effective rule formation. See Trafton and Reiser (1993) for a brief review of this issue. Despite these benefits, the evidence from worked-example research indicates some limitations of example-based learning.

For example, Chi, Bassok et al. (1989) state that, in an attempt to solve a problem, *good learners* use the examples as a reference, unlike *poor learners* who re-read the examples as though to find a solution. Similarly, VanLehn and Jones (1993) revealed that *poor solvers* tend to solve the problem analogically instead of solving the problem on their own, hence this prevents them from discovering gaps in their knowledge whereas *good solvers* use analogy sparingly as an aid to gap filling. In addition, Ross (1987) describes novice problem solving according to an example-analogy view. In an attempt to make an analogy between the new and an earlier problem, Ross argues that novices tend to rely on superficial similarities of the

problems to set up the connections between problems due to a lack of understanding of the problem structure. They also experience an “illusion of understanding” when learning from worked-examples (Renkl, 1999). In addition, they often use examples in a suboptimal way (Reimann & Schult, 1996) and fail to generalise worked-examples solutions to new problems (Catrambone & Holyoak, 1989), as cited in Moreno (2006). Among others, it has been observed that novices tend to use superficial similarities of the LISP code to solve programming tasks when details of the algorithm are not understood (Weber, 1993).

2.3.1 Instructional principles from worked-example research

Atkinson, Derry et al. (2000) have proposed a framework drawn from the literature in the area of worked-example research conducted in the past, leading to instructional design principles. More specifically, they proposed factors that moderate worked-example effectiveness. These factors include intra-example features, inter-example features, and individual differences in example processing. Intra-example features are concerned with how the worked-example is designed, for example, either by employing multiple modalities in example presentation, i.e. integration of aural and visual information (Mousavi, Low & Sweller, 1995), as cited in Atkinson, Derry et al. (2000) or via subgoal labelling of worked-example solution (Catrambone, 1995). Inter-example features are concerned with how the worked-examples are sequenced and/or arranged. For example, the presentation of multiple examples via their variability in surface features for a given problem category, that is, structure-emphasising examples (Quilici & Mayer, 1996; 2002), or pairing of an example and a similar practice problem (Trafton & Reiser, 1993). A final factor concerned with example processing is through self-explanation, i.e. through prompting the learner to self-explain the worked-example (Chi, Bassok et al., 1989) or fostering self-explanation through structural

manipulation, i.e. incompleteness of the worked-example solution (van Merriënboer, 1990b; van Merriënboer & Paas, 1990; van Merriënboer & de Croock 1992). A review of instructional principles from the worked-example research and their implications for instructional design is given by Atkinson, Derry et al. (2000).

In recent years, conventional research on worked examples has been extended towards focusing on techniques to optimise cognitive load for learning from worked-examples as derived from current developments of cognitive load theory (see Paas & van Gog, 2006; Moreno, 2006).

2.4 Cognitive load and working memory

One major assumption of cognitive load theory is that human working memory has very limited capacity. During the process of learning, most of the cognitive capacity or resources are restricted by the working memory. The basic tenet of cognitive load theory is to optimise such loads so that more working memory is available for actual learning to take place. In other words, the freed resources can (in principle) be directed to the learning activities that are relevant to the process of schema acquisition and automation.

Cognitive load theory defines three different types of cognitive load, namely intrinsic cognitive load (ICL), extraneous load (ECL), and germane cognitive load (GCL) (Paas, Renkl & Sweller, 2003; Paas & van Gog, 2006). The load which places high demands on working memory because of element interactivity (i.e. elements of the task that must be processed concurrently, which are intrinsic to the task), is referred to as ICL. The load imposed by the task that is ineffective for learning or that may impede acquisition of schema

and automation) is called ECL whereas when it is effective for learning (i.e. being helpful to acquisition of schema and automation), it is referred to as GCL. See Paas and van Gog (2006) for an overview of different types of cognitive load. Cognitive load can be optimised in three different ways (Moreno, 2006). First, by decreasing the ECL, for instance, by asking learners to work on completion problems, that is, worked example solutions that have to be completed (van Merriënboer, 1990b; van Merriënboer & Paas, 1990; van Merriënboer & de Croock, 1992). For example, this can be achieved by using a fading technique by successively introducing elements of problem-solving into the examples until the learners are able to solve problems by themselves, i.e. complete example → increasingly more incomplete examples → problem (Renkl, Atkinson et al., 2000; Renkl, Atkinson & Merrill, 2003). Also, this can be achieved by introducing problem-example pairs and/or example-problem pairs as introduced by Trafton and Reiser (1993) and Reisslein, Atkinson, Seeling and Reisslein (2006).

Second, by reducing the ICL, for instance, by reducing the interacting elements of complex learning material. This can be achieved by presenting the material as isolated elements that could be processed in working memory sequentially, rather than simultaneously (Pollock, Chandler & Sweller, 2002), or by scaffolding simple-to-complex sequencing, in other words, learning materials designed to start with relatively simple task and progress toward complex tasks (van Merriënboer, Kirschner & Kester, 2003), or by presenting learners with a modular example format (Gerjets, Scheiter & Catrambone, 2004; 2006). Whereas the modular format focuses on breaking down complex solutions into smaller meaningful elements, thus allowing only a limited number of elements to be processed simultaneously in working memory, and accordingly reducing intrinsic cognitive load, the molar format treats complex solutions as the basic unit that cannot be broken down further.

Third, by increasing the GCL, such as by providing varying representations of and multiple solutions to, a problem (Große & Renkl, 2006). This can also be achieved by increasing the variability of the worked-example problems (Paas & van Merriënboer, 1994a), and making use of high contextual interference, that is, the manipulation of the practice schedule (van Merriënboer, Schuurman, de Croock & Paas, 2002). More specifically, contextual interference refers to the practice schedule being manipulated which may hinder smooth mastery of the skills being practised. Low contextual interference refers to a blocked practice schedule. Skills for solving one type of problem are practised before progressing to a next type of problem (i.e. B-B-B, A-A-A). In contrast, high contextual interference refers to random practice schedule, that is, different problems are sequenced randomly (e.g. C-A-B, B-A-C) (van Merriënboer, Schuurman et al., 2002). Some studies within the worked-example research include prompting learners to self-explain the worked-example (Chi, Bassok et al., 1989), by asking learners to provide example elaboration on writing database queries, for instance, elaborations of the connection between the conditions and actions of SQL queries (Catrambone & Yuasa, 2006), and finally by asking learners to abstract the structural features of the examples (Quilici & Mayer, 1996; 2002).

An instructional implication of worked-example design is that example-based instruction should decrease learners' use of cognitive resources on activities that are ineffective for learning and increase learners' use of cognitive resources on activities that are relevant to schema acquisition and automation, so long the learning takes place within the overall capacity of the working memory (Moreno, 2006).

The instructional effectiveness resulting from different worked-example designs that build on cognitive load theory has been widely researched. It is worth noting, however, that the majority of these studies have failed to provide consistent findings for the instructional effectiveness with regard to learning outcomes and cognitive load effects (Paas & van Gog, 2006; Moreno, 2006). Moreno's review points out some possible new directions. Among others, one is to reconsider the assumptions laid down by the cognitive load theory under the view of the studies conducted in the area worked-example research in designing the worked-examples (e.g. Atkinson, Derry et al., 2000) as well as to include individual differences as mediating factor that may be interacting with cognitive load and learning outcomes.

2.4.1 Measuring cognitive load

“Cognitive load, a multidimensional construct, represents the load that performing a particular task imposes on the cognitive system” in Paas, van Merriënboer and Adam (1994), quoted from Paas and van Merriënboer (1994b). The dimensions of cognitive load can be conceptualised with regard to *mental load* and *mental effort*, both of which affect the learner's associated *performance* (Paas, van Merriënboer et al., 1994; Sweller, van Merriënboer et al., 1998). *Mental load* is imposed by the task demands whereas *mental effort* refers the total of cognitive capacity or resources being allocated to perform a task.

It should be noted that, the perceived intensity of the mental effort being expended by learners can be regarded as the essence of cognitive load (Paas, 1992), thus it can be used as an index of cognitive load (Paas, 1992; Paas & van Merriënboer, 1994a; Paas, van Merriënboer et al., 1994). A detailed theoretical explanation of the cognitive load construct is given by Paas and van Merriënboer (1994b).

According to Wierwille and Eggemeier (1993), measurements of mental effort can be distinguished as they are, based on three different categories, namely subjective, physiological, and task- and performance-based indices (as cited in Sweller, van Merriënboer et al., 1998) and can be measured either subjectively using rating scales or objectively using task- and performance-based techniques and physiological technique (see Paas & van Merriënboer, 1994a; Paas, van Merriënboer et al., 1994; Sweller, van Merriënboer et al., 1998; van Gog & Paas, 2008).

2.4.1.1 Subjective measures

Subjective measures consist of rating scales as developed by Paas (1992). The rating scales ranging from 1 (very, very low mental effort) to 9 (very, very high mental effort) against which the participant can indicate their perceived amount of mental effort. Paas's scale is a modified version of Bratfish, Borg and Dornic's (1972) scale for measuring perceived task difficulty. Subjective ratings of mental effort have been demonstrated to be valid, reliable, and revealed good internal consistency (Paas, Merrienboer et al., 1994). Moreover, the subjective rating is frequently used by researchers and appears numerous times in the cognitive load literature (see Paas, Tuovinen, Tabbers & Van Gerven, 2003).

2.4.1.2 Task- and performance-based or secondary task performance

Task- and performance-based measures are also referred to as a dual task approach (Brunken, Plass & Leutner, 2003) or as a secondary task methodology (Paas, Tuovinen et al., 2003), which require a primary task to be performed concurrently with a secondary task, both of which use the same working memory resources.

Performance on the secondary task (i.e. to react to the colour change of the screen background) is believed to indicate the level of cognitive load imposed on working memory by the primary task (i.e. the learning activity). Performance variables include reaction time, accuracy, and error rate (Paas, Tuovinen et al., 2003). Despite the fact that secondary task performance proved to be reliable and highly sensitive (Paas, Tuovinen et al., 2003), it may still affect the performance of the primary task (Paas, Tuovinen et al., 2003; Brunken, Plass et al., 2003). In other words, the secondary task can interfere with the primary task performance, especially when cognitive capacity or resources are limited and the primary task is complex (Paas, Tuovinen et al., 2003).

2.4.1.3 Physiological techniques

Another form of cognitive load measure is to analyse the changes in the trend and the pattern of the load as reflected by physiological variables (e.g. heart rate, pupil dilation, among others) and their correlation with the learning activities (Paas, Tuovinen et al., 2003; Brunken, Plass et al., 2003). A review of various physiological techniques is not in the scope of this thesis.

2.4.1.4 Performance on transfer and efficiency measure

Another method of investigating the effects cognitive load is to measure performance on transfer (Brunken, Plass et al., 2003). Two performance criteria commonly employed are the percentage of the problem correctly solved and time taken to solve the problem. Mental effort on the task combined with a performance measure provides information on relative efficiency of instructional conditions (Paas & van Merriënboer, 1994a; van Gog & Paas, 2008).

This is the original instructional efficiency measure proposed by Paas and van Merriënboer (1993). Based on this construct, high task or test performance combined with a low mental effort (attributable to efficiency of schemas acquired as a result of instructional format) is called high-instructional efficiency. By contrast, low task or test performance combined with high mental effort is called low-instructional efficiency (van Gog & Paas, 2008; Paas, Tuovinen et al., 2003). The efficiency measure provides a more sensitive indicator of the quality of acquired cognitive schemata than just performance tests scores on their own as it takes account of the mental effort in using them (van Gog & Paas, 2008).

Many studies have used an adapted version of the original instructional efficiency measure, (see van Gog & Paas, 2008) for an overview. The adapted measure looks at perceived mental effort or effort / perceived difficulty during learning and performance in the post-tests. It should be noted, however, that the two measures are very different in terms of what they actually measured. Whereas the original measure defines instructional efficiency in terms of the learning *outcomes*, the adapted measure defines instructional efficiency in terms of the learning *process* (van Gog & Paas, 2008). A detailed explanation of this issue is given by van Gog and Paas (2008) and Paas and van Gog (2006).

Among others, Paas, Tuovinen, van Merriënboer and Darabi (2005) proposed an alternative motivational perspective that looks at the relation between mental effort and performance in the post-tests. Specifically, this construct defines instructional efficiency in terms of instructional motivation.

2.4.1.5 Time on task

Time-on-task is an important factor that has not been addressed extensively in either the measurement of cognitive load or the calculation of instructional efficiency (Paas, Tuovinen et al., 2003; Tuovinen & Paas, 2004). Time-on-task can be considered as an objective measure of cognitive load (van Gog & Paas, 2008). An efficiency measure incorporating mental effort, performance, and the time-on-task factor (e.g. Paas, Tuovinen et al., 2003; Tuovinen & Paas, 2004), where time to complete the task is not restricted, provide a more subtle measure (van Gog & Paas, 2008). The work done by Salden, Paas, Broers and van Merriënboer (2004) provides a first step in this direction. However, they proposed total training time (instead of time-on-task) as the third dimension of the efficiency formula. In Salden, Paas et al.'s study, the three-dimensional efficiency measure was computed for adaptive/dynamic task selection during training.

2.4.1.6 Measuring three different cognitive loads separately

Despite cognitive load theory's distinction between ICL, ECL, and GCL, both objective and subjective measures indicate the total cognitive load rather than its constituent elements (Paas, Tuovinen et al., 2003; van Gog & Paas, 2008). However, recent attempts have been made to measure different cognitive load types separately (e.g. Brunken, Plass et al., 2003; Ayres, 2006; DeLeeuw & Mayer, 2008; Cierniak, Scheiter & Gerjets, 2009).

For instance, Brunken, Plass et al. (2003) argued that secondary task measures were sensitive to detecting variations in extraneous cognitive load and they have demonstrated that the approach was feasible. Brunken, Plass et al., further state that, to validate differences in cognitive load induced by different instructional strategies, both the primary task and

secondary task performance should be measured simultaneously within the same experimental setting. On the other hand, Ayres (2006) found that subjective measures were sensitive to detecting variations in intrinsic cognitive load within tasks.

DeLeeuw and Mayer (2008) reported that different measures of cognitive load were sensitive to detecting intrinsic, extraneous and germane cognitive load separately. In particular, they found that *reaction time* to the secondary task was the most sensitive to indications of extraneous cognitive load, *effort* ratings during learning were the most sensitive to indications of intrinsic cognitive load, and *difficulty* ratings after learning were the most sensitive to indications of germane cognitive load.

Cierniak, Scheiter et al. (2009) used subjective rating scales with a labelled six-point scale (from 1 - not at all, to 6 - extremely) to measure different load types, namely intrinsic, extraneous, and germane separately. Specifically, the intrinsic cognitive load scale asked “*How difficult was the learning content for you?*” The extraneous cognitive load scale asked “*How difficult was it for you to learn with the material*” as adopted from Kalyuga, Chandler and Sweller (1998). The germane cognitive load scale asked “*How much did you concentrate during learning?*” as adopted from Salomon (1984). The validity of the subjective ratings was proved to be successful in their study. However, their approach to measuring different kinds of cognitive load using subjective ratings is questionable. A closer look shows that these questions (as described above) are very much related. It is uncertain as to whether learners will really be able to answer these questions by introspection since it is hard to distinguish one question from the other two. Hence, other means of measuring different kinds of cognitive load using subjective ratings are clearly called for.

2.5 Cognitive and learning styles: effects on programming performance

2.5.1 Learning styles models

The distinction between cognitive styles and learning styles is subtle. For instance, Messick (1984) describes cognitive styles as “consistent individual differences in organizing information, and processing both information and experience” as quoted by Bishop-Clark (1995 p. 242). Keefe (1979) define learning styles as “characteristic cognitive, affective and psychological behaviours that serve as relatively stable indicators of how learners perceive, interact with, and respond to the learning environment” as quoted by Felder and Spurlin (2005 p. 104). Based on this background, both cognitive styles and learning styles will be treated under the umbrella of a similar theoretical construct. Several different, well-known learning style models are available in the literature: these include the Kolb's Experiential Learning model (Kolb, 1984) and the Felder-Silverman model (Felder & Silverman, 1988). Each model proposes a different classification of learning styles.

The Kolb's Experiential Learning model suggests four different learning styles which are based on a four-stage learning cycle, each of which corresponds to four modes of the learning process, namely **C**oncrete **E**xperience, **R**eflective **O**bservation, **A**bstract **C**onceptualization, and **A**ctive **E**xperimentation. The learning styles are the Diverger (CE/RO), the Assimilator (AC/RO), the Converger (AC/AE), and the Accommodator (CE/AE). The Kolb's Learning Style Inventory (Kolb, 1984) measures an individual's relative emphasis on each of the four modes of the learning process as described above, and another two combination scores that specify the extent to which the individual prefers abstractness over concreteness (AC-CE) and the extent to which the individual prefers action over reflection (AE-RO).

By contrast, the Felder-Silverman model suggests four learning style dimensions, namely active/reflective, sensing/intuitive, visual/verbal, and sequential/global. Learner learning style is defined in terms of having a preference for one category or the other in each of the dimensions (Felder & Spurlin, 2005). Sensing learners are concrete thinkers, practical, and prefer facts and procedures while intuitive learners are abstract thinkers, innovative, and prefer theories and meanings. Visual learners have a preference for pictures, diagrams, and flow-charts while verbal learners have a preference for written and spoken explanations. Active learners learn by trying things out and like working in groups while reflective learners learn by thinking things through and prefer working alone or with a partner. Sequential learners are oriented toward linear thinking processes and learn in small incremental steps while global learners oriented toward holistic thinking processes and learn in large leaps. See Felder and Spurlin (2005) for a detailed explanation of these learning styles.

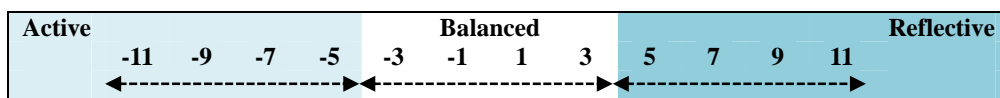


Figure 2.1: The active/reflective dimension of the ILS

The Felder-Silverman Index of Learning Styles (ILS) inventory (Felder & Soloman, n.d.) is an instrument that consists of 44 questions and is used to assess learning style preferences on the four dimensions of the Felder-Silverman model. Each dimension has 11 questions, with two possible options for answers corresponding to one or the other category of the dimension (e.g. active or reflective, see Figure 2.1) (Felder & Spurlin, 2005). The categorisation of a learner according to his/her score in a dimension can be interpreted as moderate to strong active (scores from -5 to -11) and moderate to strong reflective (scores from 5 to 11), and balanced (scores from +3 to -3).

2.5.2 Individual learner differences and programming performance

Several studies have attempted to examine the correlation between individual learner differences and programming performance. Several factors have been investigated that contributed to these differences. The factors include gender, cognitive/learning style, personality traits, problem solving or programming ability, first language, prior programming experience, prior academic performance, and instructional strategy. In our opinion, problem solving ability, cognitive/learning styles and instructional strategies appear to be a lot more promising for further research, hence the focus of this thesis.

There are several studies into the relationship between learners' learning styles and their programming performance (e.g. Byrne & Lyons, 2001; Thomas, Ratcliffe et al., 2002; Pillay & Jugoo, 2005) and/or performance in an introductory computer science course (Allert, 2004; Chamillard & Karolick, 1999). The study conducted by Byrne and Lyons (2001 p. 52) indicated that there is “a clear link between programming ability and existing aptitude in mathematics and science subjects”. Pillay and Jugoo (2005) revealed that learners’ problem solving ability has an impact on programming performance. In Pillay and Jugoo’s study, learners’ performance in Mathematics and other courses focusing on problem solving was used to measure their problem solving ability. Thomas, Ratcliffe et al. (2002) found that reflective learners scored higher than active learners on the exam portion of a course and also that verbal learners scored higher than visual learners. Similarly, Allert (2004) as well as Chamillard and Karolick (1999) found that reflective learners outperformed their scale opposites (i.e. active learners) in an introductory computer science course. By contrast, the studies conducted by Byrne and Lyons (2001) and by Pillay and Jugoo (2005) revealed that no general conclusions can be drawn from the study of learning styles with regard to learners’

achievement in programming. Among others, Carmo, Gomes, Pereira and Mendes (2006) conducted a preliminary investigation to see if there were any correlation between learners' learning style and the way they solve the problems, but found none. The study conducted by Mancy and Reid (2004) argues that cognitive characteristic such as field dependence appears to be a crucial skill in learning to program. van Merriënboer (1988) investigated the relationship between the reflectivity-impulsivity cognitive dimension and computer programming. His study suggests that the more reflective learners tended to achieve higher program comprehension test scores. Also van Merriënboer (1990a) has examined the effects of two instructional strategies, namely program completion and program generation, on reflectivity-impulsivity. In two experiments, a compensatory model was tested on the supposed negative effects of impulsivity to see if it could be compensated for by an instructional strategy that emphasises program completion as opposed to program generation. The results of the two experiments revealed no support for this model. In experiment two (a longer duration of the course was used), the data showed that there was a relation between instructional strategy and reflectivity-impulsivity, that is to say, providing support for the preference model.

Finally, Bishop-Clark (1995) has conducted an extensive review of past studies relating cognitive style and personality traits to computer programming. The review revealed that these characteristics have failed to provide consistent findings in terms of individual differences in programming performance. The majority of these studies have measured learner achievement as a single activity (Bishop-Clark, 1995). More specifically, Bishop-Clark stated: "Computer programming has been described as an activity having separate and distinct phases: problem representation, program design, coding, and debugging " (p. 242). In

other words, perhaps certain cognitive styles and personality traits affect some phases but not others. In conclusion, the empirical literature reviewed above does not provide conclusive findings in terms of individual differences in programming performance. Moreover, in the majority of these studies, the learners' achievement in programming has been measured as single activity or *overall* performance.

2.6 Related work

2.6.1 Example-based learning systems

2.6.1.1 Example-based Programming System (EBPS)

EBPS (Neal, 1989) has a simple menu-based interface for accessing the examples. EBPS uses a template approach and programming involves reuse of code. The system was developed based on the fact that novice programmers tend to either reuse previously written code or reuse code from books and manuals when coding their programs. The system was tested on twenty-two undergraduate and graduate learners with various levels of programming experience. The subjects were asked to write a program to compute change i.e. to give quarters, dimes, nickels and pennies as change for an integer value. Only six, out of twenty-two subjects did not utilize the example facility. Half of the remaining sixteen subjects made the most of the examples before and during the use of the main editing window whereas the others used them only during editing. Examples were used by learners in two fundamental ways: (1) for help with syntactic structure (2) for comparing their program to an example to verify the difference that caused the errors.

2.6.1.2 Example Tool (ET)

ET (Bowles & Robertson, 1994) is a system that uses an *approximate specification* language (AS language) as an intermediate language to support browsing examples in a database. The language is intermediate between a problem statement and the solution code in a way that makes the links between them become more apparent. Bowles and Robertson (1994) claimed that the approach addressed the two major pitfalls in analogical problem solving, namely “failure to find an appropriate example” (p. 2) and “failure to adapt appropriate features in the example solution” (p. 3). However, the system has no means to permit learners to add new examples to the database in a way that allows the learners to reflect on their learning experience. Taking this into consideration, according to Bowles and Robertson (1994), it would then be necessary to provide a means to check the learner’s solution is correct to be worth keeping it.

2.6.1.3 Episodic Learner Model Programming Environment (ELM-PE)

ELM-PE (Weber, 1993) is an intelligent programming environment supporting novices learning LISP by making use of analogies. ELM-PE was based on the premise that novices tend to use examples and reminders of solutions to previous problems to solve the current problem. ELM-PE presents reminders and analogies to the learner when errors are detected in the learner’s code or if the code is detected as a suboptimal solution. Also when the learner asks for help with writing the code, ELM-PE responds by giving analogies to similar past problems or examples. An explanation-based retrieval method (EBR) is used within ELM-PE to retrieve examples and reminders from an individual episodic case base, i.e. the Episodic Learner Model (ELM). The example selection mechanism is collaborative and is based on both the learner and the system: (1) the learner’s navigation through examples and (2)

system-suggested examples by taking into consideration the individual learner model. ELM-PE was tested with ten novices and ten advanced programmers. An experiment was conducted to test the hypothesis that novices have difficulty in identifying suitable analogies, (see Pirolli & Anderson, 1985). The results obtained in Weber's study supported the hypothesis, that novices prefer examples which are superficially related and tend to neglect structurally related examples.

2.6.1.4 Episodic Learner Model Adaptive Remote Tutor (ELM-ART)

ELM-ART (Weber & Brusilovsky, 2001) is a WWW-based version of introductory LISP course based on ELM-PE. An experiment was conducted to determine how well learners can learn with a web-based educational system. ELM-ART also provides the same collaborative mechanism for example selection as its precursor, however with slightly improved navigation, i.e. with adaptive annotation. Another characteristic of ELM-ART is that it provides 'live examples' of LISP expressions and problem-solving examples. 'Live examples' can be executed by way of a stepping mode, a visual step-by-step execution. Weber and Brusilovsky (2001) compared the results of learning with ELM-ART (tested with twenty-three learners) to the previous results from learning with ELM-PE (tested with twenty-eight learners) and reported that web-based educational systems can be as effective as Intelligent Tutoring Systems. From the feedback of users learning with ELM-ART, Weber and Brusilovsky (2001 p. 378) noted that "the adaptive examples were used not very often, however, in cases when examples were used they turned out to be very helpful". Both ELM-PE and ELM-ART have addressed one of the limitations of ET by providing learners with *reminding* to their solutions to past problems.

2.6.1.5 WebEx

WebEx (Brusilovsky, 2001), is a web-based tool for exploring tutor-explained programming examples and adopted a *dissection* method (Kelley & Pohl, 1996) of explained examples as used in programming textbooks. The exploration strategies differ in the details of the textual explanation of the examples which is sensitive to learner's current knowledge (Gomez-Albarran, 2005).

2.6.1.6 Structural Example-based Adaptive Tutoring Systems (SEATS)

SEATS (Davidovic, Warren et al., 2003) is a system that teaches JavaScript programming language by presenting examples side-by-side and highlighting their structural components using a colour-coding scheme. This technique lessens the worked-example problem of mapping by surface features. SEATS provides adaptive presentations based on a learner model. SEATS was tested with 117 learners learning recursion. The results show that using an (1) adaptive presentations mechanism with (2) the structural example-based feature increases the speed of learning and the learning gains are greater when compared with the features that are used alone, (i.e. either the adaptive mechanism or highlighting structural features) or when both the features are absent.

2.6.1.7 CORT (Code Restructuring Tool)

CORT developed by Garner (2003; 2007) supports the completion method of learning programming. The CORT interface consists of two windows. Whereas the right window contains the part-complete program, the left window contains lines of code to be used in the part-complete task. To complete the task, learners move several lines of code between the windows

and rearrange these lines in the right window. The results of initial evaluation of the CORT tool suggest that there was no difference between CORT and non-CORT learners in the performance of a final examination. However, the CORT learners took less time to solve problems and required less help.

2.6.1.8 Bridge

Bridge (Bonar & Cunningham, 1988) is an intelligent tutorial environment for novice programmers. Bridge is intended to provide an interactive feedback to learners' programming activities. Apart from finding and reporting errors, Bridge also allows learners to talk about their designs and partial programs via informal natural language dialogues. Fundamental to Bridge design are programming plans. The studies of novice programmers conducted by Soloway & Ehrlich (1984) have proved that programming plans play a crucial role to their success in program comprehension. Programming plans provide a *bridge* between informal language and programming language code. That is, Bridge allows learners to move successively from informal language descriptions of the solution through a plan specification to programming code.

In conclusion, most of the example-based systems described above have been developed for relatively narrow experimental purposes. Apart from the work of Davidovic, Warren et al. (2003), it is not clear whether these systems have been sufficiently evaluated against broader learner learning outcomes including transfer. If so, what is the impact of these systems on learner learning and problem-solving ability?

2.6.2 Cognitive load theory as a basis for the instructional design of a programming course

Further studies have embedded instructional methods derived from cognitive load theory in example-based learning. For instance, van Merriënboer (1990b), van Merriënboer and Paas (1990), van Merriënboer and de Croock (1992) suggested completion tasks that present the learners with worked-example solutions that have to be completed. The strategy forces the learners to be more reflective in studying the incomplete worked-examples provided in the completion assignment or else they cannot solve the completion tasks correctly (van Merriënboer, 1990a). Several experiments reported that a completion strategy is superior than a generation strategy (van Merriënboer, 1990b; van Merriënboer & de Croock, 1992). Chang, Chiao et al. (2000) continued this line of research by developing a programming learning system using a completion strategy involving a template technique. The system was tested with 45 high school freshman learning the BASIC programming language. The learners were randomly divided into two different groups, i.e. control group and experimental group. The results show that the learners in the experimental group who used a completion strategy performed better than the learners in the control group who studied programming by themselves, especially with respect to the knowledge of applying statements and designing programs. Finally, Trafton and Reiser (1993) introduced pairing technique, called example-problem pair that presents learners with an example followed by a matched practice problem.

Within the broader programming education literature, some studies have used cognitive load theory as a basis for the instructional design of a programming course. For example, Caspersen and Bennedsen (2007) proposed an instructional design for an introductory object-oriented programming course by adopting several theories, including cognitive load theory as

well as through deploying a well-known strategy called faded guidance, proposed by Renkl, Atkinson et al. (2000) and Renkl, Atkinson et al. (2003). Similarly, Gray, Clair et al. (2007) proposed a fading worked example strategy for teaching an early programming course. Also Yousoof, Sapiyan et al. (2007) proposed a model of measuring cognitive load during programming instruction and this model was used in an intelligent tutoring system to provide adaptive support for learning programming. On the one hand, it remains unclear whether the proposed mechanisms describe above have been sufficiently evaluated against learner learning outcomes and transfer using valid and reliable measures according to the cognitive load theory.

2.7 Conclusion

This chapter has briefly reviewed the literature that covers related interdisciplinary areas of research from analogical problem solving and transfer to worked-example research, and also research done in the area of the cognitive load theory as well as learning styles. The chapter also presented related work within the domain of programming education. Lastly, the chapter presents Table 2.1 that provides a review of worked-example design and strategy summarised within the context of instructional principles derived from the worked-example research and/or cognitive load theory-inspired research.

Table 2.1: A review of the worked-example design and strategy

Strategy/technique	Implication	Worked-example instructional design	Effects on learning
<p><i>Structure-emphasising technique</i> (Quilici & Mayer, 1996; 2002) - Statistics</p> <p><i>Structural example-based format</i> (Davidovic, Warren et al., 2003) – Programming domain</p> <p>Learners engage in the process of abstracting the structural features from superficial features of the examples.</p>	<p>Structure-emphasizing technique is effective as it “demonstrates to students that a reliance on surface features does not work” (Quilici & Mayer, 2002 p. 339).</p> <p>Problems of similar structure that share similar superficial features would help learners in categorising the problem types, and consequently assists them in solving the problems by applying appropriate method (Ross, 1989b), as cited in Atkinson, Derry et al. (2000).</p> <p>“Superficial features influence the retrieval of analogous cases in that they determine to a large degree what the problem solver perceives as similar problems” (e.g. Ross, 1987) as cited in Reimann and Schult (1996 p. 126).</p> <p>“As students become more able and confident, they could weaned away from their reliance on superficial similarities until they are able to categorize the problems by structural aspects only” (Ross, 1989b p. 464) as quoted by Atkinson, Derry et al. (2000).</p> <p>Promote structure-based schema construction (Quilici & Mayer, 2002).</p> <p>Increase GCL.</p>	<p>Encourage learners to search for commonalities between examples’ structures through <u>comparison</u> (VanLehn, 1996; Cummins, 1992; Ross & Kennedy 1990) as cited in Davidovic, Warren et al. (2003).</p> <p>Use surface features strategically to encourage search for deep conceptual structure (Atkinson, Derry et al. 2000).</p> <p>Encourage <i>structural awareness</i> by instruction (Quilici & Mayer, 2002).</p> <p><u>Question arises:</u></p> <p>What are the structural features in programming problem solving? In other words, what are programming <i>word problems</i>?</p>	<p><u>Pros</u> Learners are able to categorize the problems structurally, likely to solve the problems by generalisation, promote “structural awareness” therefore structure-based schema (Quilici & Mayer, 2002 p. 326).</p> <p><u>Cons</u> Can learner perform structural comparison between examples when they have lack of basic conceptual understanding of the domain knowledge?</p> <p>Learners do not spontaneously recognise the problem’s structure.</p>

Strategy/technique	Implication	Worked-example instructional design	Effects on learning
<p><i>Completion problem</i> (van Merriënboer, 1990b; van Merriënboer & Paas, 1990; van Merriënboer & de Croock, 1992; Chang, Chiao et al., 2000; Garner, 2003; 2007) - Programming domain</p> <p><i>Incomplete solution steps</i> (Stark, 1999)</p> <p>These strategies present learners with worked-example solutions that have to be completed.</p>	<p>Working on problem at the same time as referring worked-example results in working-memory overload. Completion problem provides an alternative approach to counteract this problem by combining the strong points of both the worked-example and the conventional problem solving tasks (Sweller, van Merriënboer et al., 1998; van Merriënboer, Kirschner et al., 2003).</p> <p>With regard to the construction of new programs, completion group outperformed generation group (van Merriënboer, 1990b) in Sweller, van Merriënboer et al. (1998).</p> <p>Incomplete example fosters self-explanation, hence the transfer of learned incomplete solution materials (Stark, 1999) as cited in Renkl, Atkinson et al. (2000).</p> <p>Learners who received completion strategy required significant less help and spent less time to solve problems than learners who have <i>conventional</i> programming exercise (Garner, 2003).</p> <p>Decrease ECL and subsequently increase GCL.</p>	<p>Part of the solution steps left for learners to complete should be carefully considered (Sweller, van Merriënboer et al., 1998).</p> <p>Incomplete examples ought to contain enough clues to guide the learners in their completion task (van Merriënboer & Paas, 1990) in Garner (2003).</p>	<p><u>Pros</u> Learners are able to reason about the relation between solutions steps that are left out for them to complete.</p> <p>Promote schema acquisition and transfer performance (Sweller, van Merriënboer et al., 1998).</p> <p>Fosters “mindful abstraction” (van Merriënboer & Paas, 1990 p. 279).</p> <p><u>Cons</u> Not suitable for learners who just beginning to learn programming.</p> <p>The technique has not incorporated dynamic fading component (see Renkl, Atkinson et al., 2000).</p>

Strategy/technique	Implication	Worked-example instructional design	Effects on learning
<p><i>Example-problem pairs</i> (Trafton & Reiser, 1993) - Programming domain</p> <p><i>Example-problem, problem-example</i> (Reisslein, Atkinson et al., 2006) - Electrical circuit analysis</p> <p>The technique presents learners with an example followed by a matched practice problem.</p>	<p>“The most efficient way to present material to acquire a skill is to present an example, and then a similar problem immediately following” (Trafton & Reiser, 1993).</p>	<p>To presents examples in close proximity to practice problems (Atkinson, Derry et al., 2000).</p> <p>Need to consider how examples and matched practice problems should be selected and intermixed (Atkinson, Derry et al., 2000).</p>	<p><u>Pros</u> Learners took less time to solve problem and produce more accurate solutions (Trafton & Reiser, 1993).</p> <p>Learners with low prior knowledge benefited from example/problem pair. In contrast, learners with high prior knowledge benefited from problem/example pair (Reisslein, Atkinson et al., 2006).</p> <p><u>Cons</u> Abrupt transitions from studying example in initial stages of cognitive skill acquisition to solving problems in later stage (Renkl, Atkinson et al., 2000).</p>
<p><i>Fading technique</i> (Renkl, Atkinson et al., 2000) - Physics domain (Renkl, Atkinson et al., 2003) - Mathematics/probability (Gray, Clair et al., 2007) - Programming domain</p> <p>The technique successively introduces elements of problem-solving into example until learners able to solve problems by themselves (i.e. complete example → increasingly more incomplete examples → problem) (Renkl, Atkinson et al., 2000).</p>	<p>Fading technique fosters learning, at least <u>near transfer</u> performance. This effect mediated by fewer errors under the fading conditions when compared to traditional method of example-problem pairs (Renkl, Atkinson et al., 2000).</p> <p>Decrease ECL.</p>	<p>As a method of fading out the worked-example solution steps, employ either backward or forward fading (Renkl, Atkinson et al., 2000).</p> <p>Use of fading <u>component</u> that allow for smooth transition from scaffolded problem solving to unaided problem solving (Renkl, Atkinson et al., 2000).</p> <p><u>Questions arise:</u> Which method of fading is more appropriate for programming domain? What is the fading component?</p>	<p><u>Pros</u> Fading technique foster learning.</p> <p>Learners likely to produce fewer errors (Renkl, Atkinson et al., 2000).</p> <p><u>Cons</u> Fading effect is restricted to near transfer performance (Renkl, Atkinson et al., 2000)</p> <p>Note: To address this, Renkl, Atkinson et al. (2003) introduce combined fading with the introduction of self-explanation prompts designed to fosters both near- and far-transfer performance.</p>

Strategy/technique	Implication	Worked-example instructional design	Effects on learning
<p><i>Modular example format</i> (Gerjets, Scheiter et al., 2004; 2006) - Mathematics/Probability</p> <p>The modular format focuses on breaking down complex solutions into smaller meaningful elements. The molar format treats complex solutions as the basic unit that cannot be broken down further.</p>	<p>This format reduces task-intrinsic, therefore, frees cognitive resources for germane activities to take place, for instance example elaboration (Gerjets, Scheiter et al., 2006).</p> <p>Decrease ICL.</p>	<p>Design an example in such a way that both <i>structural problem features</i> and <i>solution procedures</i> are treated as individual unit (Gerjets, Scheiter et al., 2006).</p>	<p><u>Pros</u> Structural problem features and solutions procedures that are broken down into smaller meaningful units can be easily conveyed separately thus enhance learning (Gerjets, Scheiter et al., 2006).</p>
<p><i>Subgoal learning</i> (Catrambone 1998; Atkinson & Catrambone, 2000) - Statistics</p> <p>“A subgoal denotes a meaningful conceptual piece of an overall solution procedure” (Atkinson & Catrambone, 2000).</p>	<p>Examples that are labelled or segmented will induce learners to self-explain why/how the steps go together, that is, to describe the purpose of steps (Catrambone, 1998; Atkinson & Catrambone, 2000)</p> <p>Foster “generalizations across problems in a domain” (Atkinson & Catrambone, 2000).</p> <p>Facilitate transfer.</p>	<p>Example should highlight a problem’s subgoal structure using structural manipulations, such as either the use of solution step labels or visually isolating parts of example (Atkinson & Catrambone, 2000).</p>	<p><u>Pros</u> Able to reason about why/how the steps work together (Atkinson & Catrambone, 2000).</p> <p>Foster generalisation, learners able to solve novel problem thus likely to increase transfer performance (Atkinson & Catrambone, 2000).</p>

Strategy/technique	Implication	Worked-example instructional design	Effects on learning
<p><i>Multiple representations (or multiple solutions) to worked example</i> (Große & Renkl, 2006) – Mathematics/Combinatorics and probability</p> <p><i>Multiple example</i> (Scheiter, Gerjets & Schuh, 2004; Scheiter & Gerjets, 2005). - Statistics</p>	<p>Learning from multiple solution methods can cause cognitive overload as learners has to mentally integrate information from disparate sources (Große & Renkl, 2006) – a phenomenon known as split attention effect (Tarmizi & Sweller, 1988).</p> <p>Moreover, this technique results in redundancy effect (Sweller, van Merriënboer et al., 1998) as structural features of every example is repeatedly presented within a problem category (Scheiter & Gerjets, 2005).</p> <p>Multiple representations (or multiple solutions) of worked-examples do not necessarily have positive effects on learning (see de Jong, Ainsworth, Dobson, van Der Hulst, Levonen & Reimann, 1998) as learners do not spontaneously distinguish interrelations between different representations (Van Someren, Boshuizen, de Jong & Reimann, 1998), as cited in Große and Renkl (2006).</p> <p>Multiple solution methods can foster learning, but do not necessarily do so. Multiple solutions decrease anticipation and even distract learners from noticing coherence between them (Große & Renkl, 2006).</p> <p>Multiple examples may be helpful for schema acquisition if learners studied the examples thoroughly and when they are guided to compare examples (Scheiter & Gerjets, 2005; Scheiter, Gerjets et al., 2004)</p>	<p>With respect to worked example design, three factors should be taken into consideration, since these factors might influence learning from multiple solutions, i.e. the <i>context condition</i> under which learning from multiple solutions are effective, different kinds of <i>multiplicity</i>, and the <i>learning goals</i> (Große & Renkl, 2006).</p> <p>Encourage learners to compare multiple solutions in order to optimise learning potential and by prompting self-explanations (Große & Renkl, 2006).</p> <p>Requires optimal learning conditions (Scheiter & Gerjets, 2005; Scheiter, Gerjets et al., 2004).</p> <p><u>Question arises:</u> To what extent this strategy effect schema acquisition and transfer?</p>	<p><u>Pros</u> Foster the process of abstraction and generalisation (Ainsworth, 2006) as cited in Große and Renkl (2006).</p> <p><u>Cons</u> Can learner perform example comparison when they have lack of basic conceptual understanding of the domain knowledge?</p> <p>Examples comparison can cause cognitive overload, i.e. split attention effect (Tarmizi & Sweller, 1988) and redundancy effect (Sweller, van Merriënboer et al., 1998).</p> <p>Learners do not spontaneously spot the interrelations between multiple examples (Van Someren, Boshuizen et al., 1998), as cited in Große and Renkl (2006).</p> <p>Decrease learners' anticipation (Große & Renkl, 2006).</p> <p>Requires optimal learning condition (Scheiter & Gerjets, 2005; Scheiter, Gerjets et al., 2004).</p>

	<p>Comparing multiple examples within a problem category foster two processes of <u>abstraction</u> (i.e. identify commonalities and differences between example) - (Scheiter & Gerjets, 2005; Scheiter, Gerjets et al., 2004).</p> <p>It should be noted that this technique do not necessary contributes to schema acquisition (Scheiter, Gerjets et al., 2004).</p>		
--	--	--	--

Chapter 3 Research questions and hypotheses

3.1 Introduction

Previous evidence in the area of learning styles research has discovered, among other things that (1) learners differ in the ways they perceive and process information, as well as respond to and interact with their learning environment (Keefe, 1979) in Felder and Spurlin (2005); (2) learners with low working memory capacity tend to prefer an active style of learning, on the other hand, learners with high working memory capacity tend to prefer a reflective style of learning (Graf, Lin et al., 2008); (3) reflective learners perform better than active learners in an introductory computer science course and/or in programming performance (e.g. Allert, 2004; Thomas, Ratcliffe et al., 2002; Chamillard & Karolick, 1999; van Merriënboer, 1988); and (4) there is an unclear relationship between reflection/impulsivity and teaching methods (van Merriënboer, 2009)¹.

In this chapter, we describe an exploratory pilot study to explore the context and factors in which learners make use of worked-examples² taking into account learners' learning styles. The chapter begins with a discussion of the pilot study. Next, the chapter notes problems associated with learning from example programs as derived from the findings of the pilot study as well as drawn from the literature. It then makes suggestions with regard to the design of a worked-example strategy and finally concludes with the research questions along with research hypotheses explored in the body of this thesis

¹ Personal communication, 14/11/2009

² Also referred to as an example or an example program in this pilot study

3.2 An exploratory pilot study

The pilot study was carried out at the Faculty of Computer Science and Information Technology, University of Malaya from the 28th of July to the 4th of August 2008. The pilot study was designed with the intent to uncover the following issues for later investigation. These included but were not limited to the following: (1) to identify the contextual and other factors that influence learners' use of example programs, i.e. from learning via examples to solving a programming task, more specifically (2) to understand learners' behaviour within such a context and the consequence of this behaviour on their programming problem solving, taking into account individual learning styles, (3) to gain some insight into various aspects of learners' possible difficulties in learning via example programs, and as a final point, (4) to gain experience of the target population, especially learners who were characterised as active or reflective. Specifically, the pilot study was conducted to explore the following hypothesis:

HAI: Learners make little use of available example programs. However, for those who make better use of example programs, then there is a clear relationship between individual learning style and the way they approach solving a programming task.

3.2.1 Method

3.2.1.1 Learning style inventory

The Index of Learning Styles (ILS) inventory (Felder & Soloman, n.d.), available at <http://www.ncsu.edu/felder-public/ILSpage.html>, recently accessed on 19/8/2011, was administered to the participants so as to evaluate their individual learning style preferences.

3.2.1.2 Participants

The pilot study involved 22 undergraduate students, the majority of whom were first year undergraduate students undertaking the WXES1114 course (Programming 1) in Semester 1 (2008/2009) and who had no prior programming background. A small number of students repeating the course in that semester were also participants. The participants were given RM10 as an incentive for taking part in this pilot study. A total of 22 participants responded to the ILS questionnaire ($Mean = -.73$, $SD = 4.15$). Table 3.1 shows the statistical data for the ILS scores.

Table 3.1: Frequency for the ILS scores

ILS scores	Frequency	Percent
-7	1	4.5
-5	4	18.2
-3	7	31.8
-1	1	4.5
1	2	9.1
3	4	18.2
5	1	4.5
7	2	9.1
Total	22	100.0

Figure 3.1 shows the distribution of ILS scores for the Active/Reflective dimension, which was slightly skewed toward the Active end of the continuum.

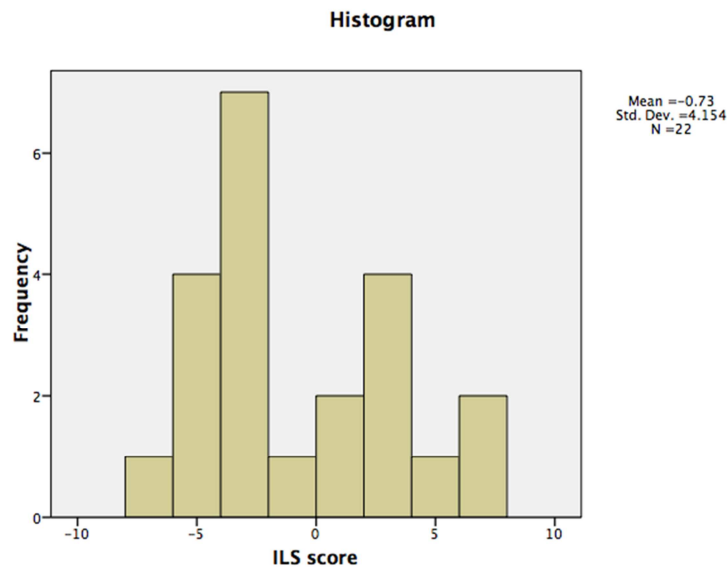


Figure 3.1: Distribution of preferences for Active (left end) / Reflective (right end) on the ILS dimension

Table 3.2 shows the preferences of the students in three categories: moderate to strong active (scores from -5 to -11), moderate to strong reflective (scores from 5 to 11), as well as balanced (scores from +3 to -3). These data were compared with the results compiled by Felder and Spurlin (2005) from the past studies and with that of the work done by Graf, Viola, Kinshuk and Leo (2006). The data from this pilot study were consistent with those obtained in the past studies. This indicates that the sample data of this study was representative.

Table 3.2: Preferences for learning styles on the Active/Reflective dimension of the ILS

	Moderate to strong active	Balanced	Moderate to strong reflective
This study (n = 22)	22.7%	63.6%	13.6%
A (n = 183)	24%	61%	15%
B (n = 207)	24%	61%	15%
C (n = 87)	27%	58%	15%

Note: A (Felder & Spurlin, 2005); B (Graf, Viola et al., 2006); C (Felder & Spurlin, 2005)

3.2.1.3 Procedure

Each participant was given a description of the pilot study and asked to sign a consent form. The study consisted of programming tasks to be solved in Java. For the purpose of the study, the tasks were designed around the topics of the MOD and DIV operators. These tasks were concerned with the three main program development stages: designing a solution, coding the solution, and finally debugging and/or executing the program solution.

The participants were given 3 problem sheets (i.e. programming tasks) and they could start working with any of the problems, in no particular order. Each problem sheet consisted of a description of the problem to be solved and a blank space for them to write their solution. These tasks were accompanied with a booklet of example programs that illustrated the use of MOD and DIV operators. One of the example programs was directly related to the program modification task, i.e. Task 3, see Figure 3.2. These tasks involved the use of a Java development editor.

Some of the early participants who took part in this pilot study received a booklet consisting of 5 example programs, two of which included a CASE statement. However, one of the example programs written with a CASE statement had to be changed to an IF/ELSE statement while the other one was simply excluded from the booklet after the experimenter has been informed that the majority of the participants have yet to learn the CASE statement. So, for the later participants, the booklet consisted of 4 program examples, all written with an IF/ELSE statement. This oversight, however, did not affect the analysis of the data as the main goal of this study was to observe the extent to which the participants were able to use MOD and DIV operators in solving the programming tasks.

1. To *write* a program that calculates a change due and determine how many RM notes (i.e. RM1, RM5, or RM10) a customer should receive³.
2. To *write* a program that converts 24 hour format into the equivalent time in AM/PM³.
3. To *amend* a program (i.e. example program 4 in the booklet) that will assign a “+” or “-“ sign after a letter grade, based on the last digit of scores. For example 98 is A+ and that 93 is A-⁴.

Figure 3.2: The programming tasks

Time-on-task was not strictly controlled. However the participants were told to finish the three problem sheets within a time frame of 50 minutes or so. The experimenter encouraged participants to attempt all three tasks. The experimenter observed the participants solving the tasks and took notes about what they did and what they said. Screen video capture⁵ was used to record the coding activity.

As we employed a semi-structured observational study, the participants were merely told about the booklet for them to look at and the overall time given to solve the tasks, however they were not informed about how to approach solving a programming task (i.e. step-by-step approach to program development and coding). The experiment took approximately 50 minutes (per participant). Finally, the participants were asked to complete the ILS online questionnaire (i.e. after each individual session).

³ Unknown source

⁴ Problem taken from Practical C programming by Oualline, Steve, 3rd Edition, O'Reilly (1997)

⁵ Freez screen video capture, available at <http://www.smallvideosoft.com/screen-video-capture/> (recently accessed on 20/8/2011).

3.2.1.4 Coding observational data

Table 3.4 presents the coding scheme, consisting of 14 categories (i.e. A1-A4, B1-B6, and so on) for noting and analysing the observational data. Note that, categories A1-A4 were obtained from the study conducted by Chi, Bassok et al. (1989), where these were the factors observed in their study on how learners study and use examples while they solve problems. Note also that category C2 was obtained from the study conducted by Neal (1989). The coding scheme was indirectly drew on the study conducted by Garner (2007) who identified five distinct levels of cognitive strategy in relation to the usage of the CORT system to support learning programming. The coding scheme also included observational behaviour data directly relevant to this pilot study, e.g. copy lines of code from an example program to an editor and execute it to check the output. In general, the recording and coding process involved three distinct steps:

1. Assign a category (i.e. A1, or B2, etc.) to each piece of data observed (with reference to the coding scheme described below).
2. Calculate number of times such a category occurred (i.e. recurrence rate) within specified period of time (i.e. 50 minutes).
3. Based on the calculated recurrence rate, assign a specific value, see Table 3.3. The values, as described in the table are then used in analysis of correlation to investigate a relationship between the factors assessed and ILS scores.

Table 3.3: 4-point rating scale for observational data analysis

	Rate of recurrence	# Values
Never	0	1
Rarely (i.e. hardly ever)	1 - 3	2
Occasionally (i.e. sometimes)	4 - 6	3
Frequently	≥ 7	4

Note: # the values and their associated meaning are to code into ordinal variables

Table 3.4: Coding categories and scheme for analysing observational data

Coding categories		Coding scheme	
What the learner do with the example programs			
A1: Look at example programs superficially (on the surface).		Scan or glance through example programs.	
A2: Refer or read example programs with understanding / self-explain example programs.		Shows some evidence of carefully read, study, and self-explain example program(s): <ul style="list-style-type: none"> - Study the algorithm and look at the output (i.e. sample run) of example program(s). - Copy lines of code from an example program to an editor and execute it to check the output. - Compare between example programs. - Compare a program solution with an example program so as to determine how the program should execute / an output should be displayed. 	
A3: Refer or read example programs as if to <i>search for a solution</i> .		Shows some evidence of thoughtless approach (spontaneous): <ul style="list-style-type: none"> - Choose example program(s) in arbitrary manner. - Refer to incorrect/dissimilar example program(s). - Partially read example program(s) / re-read example program(s). - Indicates some evidence of <i>attempting</i> to solve algorithm though an example program referred to is incorrect/dissimilar. - Indicates some evidence of failing to apply analogous example program's algorithm. - As if searching for clues (surface features, i.e. time, convert, and currency). 	
A4: Refer to an example program as a <i>source of specific reference</i> .		Shows some evidence of thoughtful approach (on purpose): <ul style="list-style-type: none"> - Thoughtfully choose example program(s). - Refer to analogous example program(s). - Carefully read example program(s). - For algorithmic guidelines. - For syntax guidelines. - For semantic guidelines (meaning of example program). - For structural guidelines (related to task / program requirement especially for Task 3; program structure / language construct i.e. placement of code, variable declaration). - To debug a program solution. 	
Point at which the learner make use of example programs			
B1: Before attempting the tasks to reflect on what the learner have learned.		- Related to A2	
B2: Understanding the programming problem / problem representation.		- Related to A2, A4	
B3: Designing a program solution.		- Related to A4	
B4: Coding a program solution.			
B5: Debugging a program solution.		- Related to A4, C1, C2, C3	
B6: After executing a program for checking solution.		- Related to A2, C4	
How example programs are used			
C1: As an <i>implementation</i> of algorithm / an <i>attempt</i> to solve algorithm.		- Related to A4 (i.e. implementation), A3 (i.e. an attempt)	
C2: To prompt / hint at syntactic, semantic (Neal, 1989), program specification, program structure (or language construct).		- Related to A4, B3, B4, B5, C3	
C3: To debug errors.		- Related to A4, B5, C2	
C4: To check solution.		- To test a program solution / an example program or to compare a program solution with an example program so as to determine how the program solution should execute / an output should be displayed.	
		- Related to A2, B6	

3.2.2 Results

In this section, the results of the pilot study are presented. The section starts with the results of a correlation analysis between learning style and the factors assessed, followed by programming tasks performance, and lastly, a summary of observations of the participants.

Table 3.5 summarises the results, represented in terms of mean *frequency* per participant, in which the factors assessed in the pilot study occurred for a given time period, i.e. 50 minutes. Note the lists of factors that influence the use of example programs.

Table 3.5: Factors assessed (that influence the use of example programs) in the pilot study

	(n = 22)	
	Mean	SD
<i>What the learner do with the example programs</i> (Chi, Bassok et al., 1989)		
A1: Look at example programs superficially (on the surface)	3.27	2.27
A2: Refer or read example programs with understanding / self-explain example programs	1.77	2.71
A3: Refer or read example programs as if to <i>search for a solution</i> .	5.00	6.64
A4: Refer to an example program as a <i>source of specific reference</i> .	6.68	6.13
<i>Point at which the learner make use of example programs</i>		
B1: Before attempting the tasks to reflect on what the learner have learned	0.82	1.92
B2: Understanding the programming problem / problem representation	2.32	2.01
B3: Designing a program solution	1.64	3.76
B4: Coding a program solution	7.09	6.74
B5: Debugging a program solution	1.64	1.99
B6: After executing a program for checking solution	0.14	0.35
<i>How example programs are used</i>		
C1: As an <i>implementation</i> of algorithm / an <i>attempt</i> to solve algorithm	6.45	6.54
C2: To prompt / hint at syntactic, semantic (Neal, 1989), program specification, program structure (or language construct)	4.41	4.59
C3: To debug errors	1.18	1.89
C4: To check solution	0.55	1.79

3.2.2.1 Correlation analysis

This section presents the results of the correlation analysis between ILS scores and factors that influence the use of example programs. When correlated with learning style scores, most of the factors assessed were not highly correlated. Only one factor, C1 showed negative correlation, and that was very weak and not significant. Table 3.6 shows the five highest positive correlations, indicated by a significant p value of small to medium.

Table 3.6: Results of the correlations between learning style scores and the factors

	<i>rho</i>	<i>p</i>
<i>What the learner do with the example programs:</i>		
A2: Refer or read example programs with understanding / self-explain example programs.	.28	0.21
A4: Refer to an example programs as a <i>source of specific reference</i> .	.40	0.06
<i>Point at which the learner make use of example programs:</i>		
B1: Before attempting the tasks to reflect on what the learner have learned.	.47	0.03
<i>How example programs are used:</i>		
C2: To prompt / hint at syntactic, semantic (Neal, 1989), program specification, program structure (or language construct).	.37	0.10
C3: To debug errors.	.28	0.20

Note: Correlation is significant at the 0.05 level (2-tailed) and highlighted in bold.

As shown in Table 3.6, as might be expected, the results show moderate correlations between B1 (also A4) and learning style scores. The two findings, with respect to the B1 and A4 factors will be further discussed in the following paragraphs. On the other hand, the results show weak correlations between A2, C3 and learning style scores; and a moderate correlation for C2. These were not statistically significant due to the small sample size.

There was a positive correlation between B1 and the learning style scores, ($\rho(22) = 0.47, p = 0.03$). This correlation indicates that the more reflective learners tended to reflect on the example programs given before starting the tasks.

Similarly, there was a positive correlation between A4 and the learning style scores, though not significant, ($\rho(22) = 0.40$). This correlation indicates that the more reflective learners tended to refer to the example programs as a source of specific reference. In agreement with the past evidence, (Keefe, 1979) in Felder and Spurlin (2005), the results of the correlation analysis show a link between learning styles and the way learners approach a task, in this pilot study's case, solving a programming task.

Note that we have conducted multiple tests using Spearman's ρ for investigating correlations between the ILS scores and several dependant variables (i.e. factors). This may have caused a Type 2 error to occur – getting a significant result by chance. To minimise the possibility of reaching a wrong conclusion, we used an unadjusted alpha value at $p < .05$ and considered it to be significant if an effect size reached **at least** a small to medium effect (e.g. .25 to .30). A non-significant result merely indicates a trend if the result reached at least the minimum criteria of an effect size. Finally, the *HAI* hypothesis was somewhat supported. Nevertheless, this hypothesis remains tentative due to the small sample size. Thus further investigation is clearly called for.

3.2.2.2 Programming tasks performance

The task scores were determined by a set of criteria (i.e. marking scheme) as described in Table 3.7. An example of a marking scheme for Task 2 was as follows:

Table 3.7: Marking scheme for Task 2

Criteria	Points
Input	1
Use of MOD and DIV operators	2
Selection statement (i.e. IF/ELSE statement)	2
Total points that could be earned	5

Note: Points varies depending on the accuracy of answer given. The selection statement was also counted in the score for it was an essential element of the solution.

Table 3.8 and Table 3.9 show the participants' programming task scores for those whose learning style preference were either moderate-active (scores from -5 to -7) or moderate-reflective (scores from 5 to 7). In general, the reflective participants scored better in the programming tasks than the active participants, and this was broadly consistent with the past evidence, (e.g. Allert, 2004; Thomas, Ratcliffe et al., 2002; Chamillard & Karolick, 1999; van Merriënboer, 1988).

Table 3.8: Results of programming task performance (active learners)

Name (not a real name)	ILS score	Score on Task 1	Time spent on Task 1	Score on Task 2	Time spent on Task 2	Score on Task 3	Time spent on Task 3
Maziah	-7	1.5	25	1	25		
Rohana	-5	2	20	3	30		
Brian	-5	2.5	33	1	14	3	3
Aziz	-5	2.5	15			4	35
Izlan	-5	3	21	1	15	3.5	14
Mean		2.3	22.8	1.5	21	3.5	17.3

Indicates that participant failed to attempt the task due to the time constraint. Programming task scores (0-5). Time spent in minutes.

Table 3.9: Results of programming task performance (reflective learners)

Name (not a real name)	ILS score	Score on Task 1	Time spent on Task 1	Score on Task 2	Time spent on Task 2	Score on Task 3	Time spent on Task 3
Lillian	5	5	29	1	21		
Peter	7	5	12	1	23	5	15
Cindy	7	5	16	0.5	12	4	16
Mean		5	19	0.83	18.7	4.5	15.5

Indicates that participant failed to attempt the task due to the time constraint. Programming task scores (0-5). Time spent in minutes.

3.2.2.3 Observation of the participants

This section presents a selection of the observations made of the participants while they were working and is also derived from the interviews held after they had finished.

- Lillian did not study and self-explain example programs prior to solving the programming tasks. She could not find any similarities between the example programs and the to-be-solved tasks.
- Peter did not self-explain example programs. He only looked at the example programs superficially (on one occasion) prior to solving programming tasks. He found that the example programs were not very useful.
- Izlan has previous programming background (C grade).
- Cindy studied and self-explained example programs prior to solving programming tasks. She found that the example programs were very helpful.
- Almost all the participants seemed to refer to the example programs rather superficially (as reported by Ross, 1987) in order to solve the programming tasks. More specifically, they referred to the surface features of the example programs without being able to recognise the common structural features between the programming tasks and the example programs (i.e. the use of MOD and DIV operators).
- Some participants said that they knew that one or two example programs in the booklet might aid them in solving the programming tasks. However they failed to

relate the tasks given to the example programs. All the participants (except one who failed to recall them) had learnt the MOD and DIV operators in class and in one of the lab assignments which required the use of these operators. Indeed, all of the example programs illustrated the use of these operators, but to little effect.

- Almost all the participants spent little time reading or self-explaining the example programs. Some of the participants used the example programs as a reference, while the rest of the participants relied heavily on the example programs as if looking for a solution (as reported by Chi, Bassok et al., 1989). One participant self-explained an example program (by copying the program to the editor, editing it, and running it) and tried to understand the algorithm, but failed to analogise, especially with regards to the use of MOD and DIV operators.
- Some of the participants preferred not to look at the example programs, the reason being that they wanted to explore and learn more without having to depend too much on the example programs. One participant believed that he could solve the tasks without even looking at the example programs.

3.3 Planning for the main experiment

In this section, we identify various aspects of learners' difficulties in learning via example programs, derived from the findings of the pilot study as well as drawn from the literature, see Figure 3.3 below.

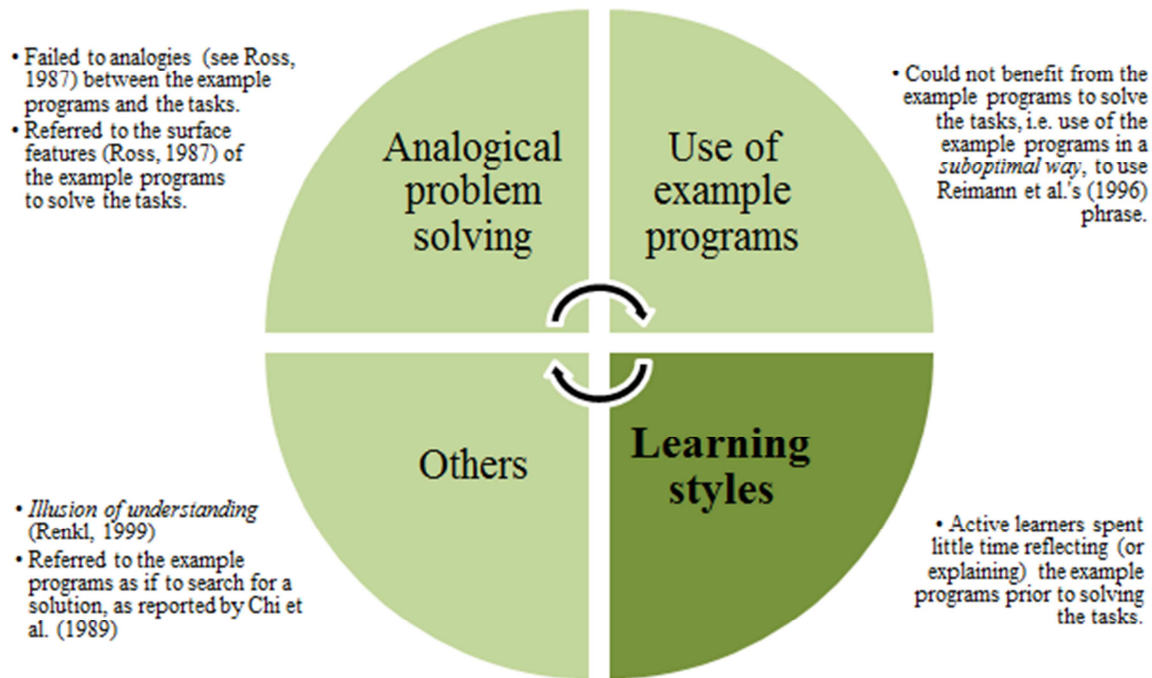


Figure 3.3: The difficulties inherent in learning via example programs

In developing the materials and hypotheses for the main experiment, we focused on the instructional principles from the worked-example research (Atkinson, Derry et al., 2000), by incorporating their three moderating factors into the design of worked-example strategy.

More specifically, we addressed the above issues (1) via intra-example features (how the worked-example is designed), i.e. the incompleteness of the worked-example solution, (2) via inter-example features (how the worked-examples are sequenced), i.e. providing example-problem pairs and/or variability of surface features within a problem category, and finally (3) via prompting to self-explain the worked-example.

We concluded that the Structure-emphasising strategy (Quilici & Mayer, 1996; 2002) and the Completion strategy (van Merriënboer, 1990b; van Merriënboer & Paas, 1990; van Merriënboer & de Croock 1992) were the two most practical strategies for incorporating the

three moderating factors into the design of a worked-example strategy, see Table 2.1 in Chapter 2 for an overview of the two strategies. We proposed a Paired-method strategy that combines a Structure-emphasising strategy with a Completion strategy to implement the example-problem pair (Trafton & Reiser, 1993) which aimed at helping active learners gain more benefit from being exposed to worked-examples, while at the same time not disadvantaging reflective learners.

The paragraphs that follow briefly discuss the interactions between the worked-example strategy and learning styles (and working memory capacity) to justify the proposed preferential model in predicting outcomes as opposed to the compensatory model. There are three main strands to the argument.

1. Effective design of a worked-example strategy does not in itself guarantee positive learning outcomes (see Atkinson & Renkl, 2007). The worked-example strategy is an instructional strategy that requires learners to “actively process the examples” (Atkinson & Renkl, 2007 p. 378) and requires “mindful abstraction” from the learners (Paas & van Merriënboer, 1990 p. 279). The benefit of the worked-example, strategy however, also depends on the learner’s willingness to expend serious effort on understanding the worked-examples (Paas, Tuovinen, van Merriënboer & Darabi, 2005). Nevertheless, it is worth noting that, in general, a strategy based on worked-examples is not necessarily suited for active learners. Research in the area of learning styles indicates that active learners learn by *trying things out* and prefer working dynamically and in a group (Felder & Spurlin, 2005), thus learning via a worked-example strategy can be inferior for active learners as they may become overwhelmed

by the strategy. In contrast, reflective learners profit from studying worked-examples as they learn by *thinking things through*, to use Felder and Spurlin's (2005) phrase. Indeed, as Atkinson and Renkl (2007 p. 377) argued "...the way in which learners' study and process examples has a dramatic impact on whether learning occurs". Note that van Merriënboer's (1990a) studies found little support for the idea that the negative effects of impulsivity (a feature of active learners) could be compensated by an instructional strategy which emphasised program completion (as described in Chapter 2, section 2.5.2).

2. To summarise the main issue, we argued that learning style is a factor in determining whether learners benefit from studying worked-examples. This is consistent with Graf, Lin et al.'s (2008) view on the *indirect relationship* between learning styles and working memory capacity. While reflective learners with high working memory capacity benefit from studying worked-examples, active learners with low working memory capacity do not - for they may become overwhelmed by the strategy, and accordingly may experience cognitive overload. Moreover, instructional design based on cognitive load theory argues for the careful utilisation of working memory capacity to encourage more effective schema construction (Sweller, van Merriënboer, & Paas, 1998).
3. The proposed Paired-method strategy is expected to **moderate worked-example effectiveness** for active learners, while not disadvantaging reflective learners. In the Structure-emphasising strategy, it is argued that less reflective activity is required as learners are guided through explanation activity by means of self-explanation prompts

and hints. In other words, extraneous cognitive load is expected to decrease as learners' attention is directed to processing elements relevant to schema acquisition. In the Completion strategy, learners are given the opportunity to apply their knowledge into practise. Similarly, it is argued that less reflective activity is needed in solving a completion task - attributable to the declarative knowledge acquired from studying the worked-example using the Structure-emphasising strategy. Accordingly, extraneous cognitive load is expected to decrease further and more cognitive resources can be allocated for actual learning. Using the Paired-method strategy, extraneous cognitive load is expected to drastically decrease when compared to either of the single strategies on their own. Consequently, more effort can be invested in processes germane to the learning activity. The next chapter (i.e. Chapter 4) deals with the detail designed of the Paired-method strategy, its theoretical assumptions as well as the rationale for the design of the strategy.

In conclusion, the expectation was that active learners would engage well with the Paired-method strategy of interacting with the worked-examples and would show learning gains approaching those of reflective learners using the same strategy. Active learners would be expected to do worse than reflective learners when exposed only to a single strategy of using worked-examples. Finally reflective learners using the Paired-method strategy would show learning gains slightly better than reflective learners using either of the single strategies on their own.

3.4 Research questions and hypotheses

The findings from the pilot study and the previous research discussed above suggested several interesting research questions. Specifically, this research was driven by the following questions: To what extent does the design of worked-example strategy foster schema acquisition and transfer? Are they mediated by individual learning style? Perhaps, one of the most important aspects to explore is the interaction between learning style and learners' cognitive load. Finally, the underlying issue is, what difference does the learning via worked-examples strategy make to the quality of the cognitive schemata acquired and to the transfer of programming problem solving skills?

To answer the research questions, the research investigated any differential effects of the different worked-example strategies on the learning *process* and *outcomes* (including transfer) as well as on the cognitive load that occur during learning, taking into account learners' learning styles. The research also investigated the relationship between learning styles and learners' working memory capacity.

With regard to the differential effects of the different worked-example strategies, the research investigated the following predictions:

3.4.1 The H1 hypothesis

Given the same amount of time on task with similar instructional content, it was hypothesised that the Paired-method strategy would lead to better learning⁶ than with either the Structure-emphasising strategy or the Completion strategy alone. No prediction was made with regard to the direct comparison between the Structure-emphasising strategy and the Completion strategy.

3.4.2 The H2 hypothesis

The Paired-method strategy would lead to better near and far transfer performance than with either the Structure-emphasising strategy or the Completion strategy alone. No prediction was made with regard to the direct comparison between the Structure-emphasising strategy and the Completion strategy.

With regard to learning styles, the research investigated the following prediction: given the same amount of time on task with similar instructional content, it was hypothesised that the Paired-method strategy would lead to better learning for active learners and that reflective learners would do no worse than with either the Structure-emphasising strategy or the Completion strategy alone. The subsequent sections present the specific research hypotheses that were addressed. Table 3.10 summarises these hypotheses.

⁶ In terms of the learning *process* (i.e. higher effort and lower difficulty)

3.4.3 The H3 hypothesis

It was hypothesised that the effectiveness of learning from worked-examples is very much dependent on individual learning styles. This assumption was made based on the work of Graf, Lin et al. (2008) who identified the relationship between working memory and learning styles. In particular, learners with high working memory capacity tend to prefer a reflective learning style. On the contrary, learners with low working memory capacity tend to prefer an active learning style.

3.4.4 The H4 hypothesis

It was expected that reflective learners who have high working memory capacity would perceive their effort (source of germane cognitive load) and difficulty (source of extraneous cognitive load) as high and low respectively, with both the Structure-emphasising strategy and with the Completion strategy. The reason being that, these strategies provide reflective learners with more opportunity for *thinking things through*, to use Felder and Spurlin's (2005) phrase, thus these instructional formats are effective for this type of learner.

By contrast, it was expected that active learners who have low working memory capacity would perceive their effort and difficulty as low and high respectively, with both the Structure-emphasising strategy and with the Completion strategy. This is because these strategies forced active learners into an uncongenial, more reflective style of learning. Neither format provides an opportunity for active learners to work with the learning material dynamically by *trying things out*, again to use Felder and Spurlin's (2005) phrase. Hence these instructional formats are ineffective for this type of learner.

However, it was expected that reflective learners would show no difference with respect to their perceived effort and difficulty with the Paired-method strategy. Also, it was expected that active learners would perceive their effort and difficulty just about or equally high and low respectively, with the Paired-method strategy like reflective learners. Therefore the Paired-method strategy would be effective for both active and reflective learners.

3.4.5 The H5 hypothesis

There should be relative merits concerning near and far transfer performance of the three worked-example strategy conditions with different learning style categories. In particular, active learners would show worse near and far transfer performance than reflective learners, in both the Structure-emphasising strategy and the Completion strategy. In the Paired-method strategy, reflective learners would show just slightly better or equal near and far transfer performance as compared to their reflective counterparts in the other two strategies. By contrast, active learners would show just about or equal near and far transfer performance like reflective learners.

Table 3.10: Research hypotheses

The H3 hypothesis					
Individual learning styles.					
High working memory capacity, learners tend to prefer a reflective learning style.				Low working memory capacity, learners tend to prefer an active learning style.	
The H4 hypothesis					
Difficulty ratings and effort ratings for different types of load, namely ECL and GCL, respectively.					
Structure-emphasising strategy		Completion strategy		Paired-method strategy	
Reflective	Active	Reflective	Active	Reflective	Active
Effort as high Difficulty as low.	Effort as low Difficulty as high.	Effort as high Difficulty as low.	Effort as low Difficulty as high.	No difference with respect to effort and difficulty as compared to their reflective counterparts in the other two groups.	Just about or equally high effort and low difficulty like reflective learners.
The H5 hypothesis					
Near and far transfer performance					
Structure-emphasising strategy		Completion strategy		Paired-method strategy	
Reflective	Active	Reflective	Active	Reflective	Active
Better near and far transfer.	Worse near and far transfer.	Better near and far transfer.	Worse near and far transfer.	Just slightly better than or equal near and far transfer to those of reflective counterparts in the other two groups.	Just about or equal near and far transfer like reflective learners.

Note: ECL = Extraneous cognitive load; GCL = Germane cognitive load

3.5 Conclusion

This chapter has briefly discussed the pilot study that explored learners' difficulty in learning via example programs and the roles that example programs and learning styles might play. It has also investigated the interaction between learning styles and individual problem solving in a programming task and tested a hypothesis, namely that **learners make little use of available example programs. However, for those who make better use of example programs, then there is a clear relationship between individual learning style and the way they approach solving a programming task** was somewhat supported. This found that learning style is a factor in determining whether learners benefit from studying worked-examples.

Chapter 4 LECSES

4.1 Introduction

In employing worked-examples¹ to teach programming effectively to both active and reflective learners, the research proposed a worked-example strategy, called a Paired-method that combines a Structure-emphasising strategy with a Completion strategy. *Structure-emphasising*, to use Quilici and Mayer's (2002) phrase is the strategy that requires learners to explain the examples' underlying plan structures. The Completion strategy requires learners to complete the example's solution where some of the code is missing (van Merriënboer, 1990b; van Merriënboer & Paas, 1990; van Merriënboer & de Croock 1992). This chapter starts with a brief introduction of the proposed Paired-method strategy. Then, the chapter presents a discussion of the theoretical assumptions and rationale for the design of the Paired-method strategy, followed by related work underlying the interface design. Next, the chapter briefly discusses the LECSES development environment, two types of web-based interfaces supporting Structure-emphasising and Completion strategies, the LECSES editor, the LECSES administrator module, and finally the conclusions.

4.2 The paired-method strategy

Table 4.1 describes the principles underlying the design of the Paired-method strategy. A central notion is the instructional principles from the worked-example research (see Atkinson, Derry et al. 2000).

¹ Also referred to as an example or an example problem, particularly when describing the interaction involved as the learner engages in learning with LECSES (i.e. section 4.3.2)

Table 4.1: A summary of the design of the Paired-method strategy

	Worked-example strategies		
Instructional principles from the worked-example research (Atkinson et al., 2000)	<i>Structure-emphasising strategy</i> (Quilici et al., 1996; 2002)	<i>Completion strategy</i> (van Merriënboer, 1990b; van Merriënboer et al., 1992)	<i>Paired-method strategy</i>
<i>Intra-example features</i>	Promote “structural awareness” (Quilici et al., 2002 p. 326) via emphasising programming plan structures in an example solution, hence self-explanation can be readily encouraged.	Incompleteness of programming plan structures, embedded in an example solution.	Complement the intra-example features of the Structure-emphasising strategy and the Completion strategy.
[#] <i>Inter-example features</i>	Variability of surface features within a problem category (Quilici et al., 1996; 2002; Gerjets et al., 2008). That is, presenting sequence of worked-example using the same Structure-emphasising strategy format.	Variability of surface features within a problem category (Quilici et al., 1996; 2002; Gerjets et al., 2008). That is, presenting sequence of worked-example using the same Completion strategy format.	Variability of surface features within a problem category (Quilici et al., 1996; 2002; Gerjets et al., 2008). That is, presenting a structure-emphasising worked-example followed by a matched problem that has to be completed - Example/problem pairs (Trafton & Reiser, 1993).
<i>Self-explain worked-example</i>	Structure-based (i.e. programming plan - Soloway, 1986) self-explanation guidance through plan-focused prompts. Largely concerned with self-explanation (and reflection) on <i>procedural</i> aspects of an example solution.	Promote self-explanations via structural manipulations. That is, partial example solution (i.e. programming plan structures) that has to be completed (Atkinson et al., 2000) or that has to be modified.	Promote learning how to construct an explanation and directs attention to mechanism, i.e. how an incomplete solution should be completed (Soloway, 1986).

Acquisition of declarative and/or procedural knowledge	Plan schemata – <i>declarative knowledge</i> only and to a lesser extent, procedural understanding.	Problem solving rules – <i>procedural knowledge</i>	An opportunity to apply <i>declarative knowledge</i> acquired from studying a worked-example using the Structure-emphasising strategy into practise through solving a completion task. Hence, acquisition of effective problem solving rules (i.e. <i>procedural knowledge</i>)
Cognitive load account	Extraneous cognitive load is expected to decrease as learners' attention is directed to processing elements relevant to schema acquisition, thus increasing germane cognitive load.	Extraneous cognitive load is expected to decrease as learners' attention is directed to processing elements relevant to schema acquisition, thus increasing germane cognitive load.	Extraneous cognitive load is expected to drastically decrease as less reflection is needed in solving the completion task - attributable to the <i>declarative knowledge</i> acquired from studying the worked-example using the Structure-emphasising strategy. Accordingly, more cognitive resources can be invested in processes germane to the learning activity.

Note: [#] The benefits of presenting pairing sequence of worked-examples using two different strategies (i.e. the Paired-method) may be readily observed when compared with a paired sequence of worked-examples using the same Structure-emphasising strategy (or Completion strategy) format.

In designing an effective worked-example strategy, we note one important point of concern, as Anderson (2009)² said, the success of the strategy (Paired-method) really turns very much on how this early stage is organised. Get this right up front. Identify what the real appropriate organisation is for student to have at this level. In addition, we formulate the Paired-method strategy based on van Merriënboer and Krammer's (1987) proposed tactics, used to design or evaluate strategies for introductory computer programming courses.

The following paragraphs discuss the theoretical assumption of the proposed approach with reference to the research literature.

4.2.1 Theoretical assumptions and rationale for the design of the Paired-method strategy

Knowledge compilation theory suggests that studying examples can help learners to construct declarative knowledge only and this view further claims that learners can acquire problem solving rules by applying this knowledge through solving a problem (Anderson, 1987), as reported by Trafton and Reiser (1993). Trafton and Reiser's finding is consistent with the theory and they found that learning is hampered when the sources of the analogous examples are not readily accessible³ to the target problems. More specifically, Trafton and Reiser claimed that "...for an example to be most effective, however, the knowledge gained from the example must be applied to solving a new problem. The most efficient way to present material to acquire a skill is to present an example, and then a similar problem immediately

² Personal communication, 29/7/2009

³ That is, analogous examples provide some kind of retrieval cue, i.e. directly accessible for solving the current problem.

following”. In brief, problem solving guided by an accessible example helps to form effective problem solving rules (Trafton & Reiser, 1993).

The proposed strategy combines the Structure-emphasising strategy with a Completion strategy to implement an example-problem pair (Trafton & Reiser, 1993). Whereas studying an example directly concerned with the acquisition of declarative knowledge, solving a problem is specifically concerned with the acquisition of procedural knowledge (van Merriënboer & Krammer, 1987).

According to van Merriënboer and Krammer (1987), an important aspect of declarative instruction is teaching *schema-like* knowledge, such as programming plans – a theory proposed by Ehrlich and Soloway (1984). Plans are one kind of generic program fragments, for example, a running total loop plan, a running total variable plan (among others). These plans represent “stereotypic actions in a program” (Ehrlich & Soloway, 1984 p. 115). The fundamental idea is that “...there is a relationship between types of problems and types of programs, and that the notion of programming plans can serve to highlight the commonalities that do exist” (Soloway, 1985 p. 171). The theory of programming plans has been applied within the programming education domain. Indeed some researchers have developed intelligent tutoring systems (BRIDGE: Bonar & Cunningham, 1988; PROUST: Johnson & Soloway, 1985) to teach programming via plans. Note that fundamental to the Paired-method design is the use of programming plans (see Table 4.1) and thus the teaching of a structured programming technique.

Self-explanation may influence the effects of worked-examples in that it “helps to create a deeper understanding of material, eventually learning good procedure” (Anderson, 2009)⁴. According to Conati and VanLehn (2000 p. 389), self-explanation refers to the process of “...generating explanations to oneself to clarify an example’s worked-out solution”. From a programming perspective, Soloway (1986 p. 851) argued that “...learning to program amounts to learning how to construct mechanisms and how to construct explanations”. Thus, a central notion of the Paired-method strategy is self-explanation, such as explaining *procedural* aspects of an example solution (i.e. programming plan structures).

Procedural instruction is primarily concerned with facilitating knowledge compilation processes that is applying acquired declarative knowledge to practice. In this context, worked-examples can be an effective means to provide such instruction (see van Merriënboer & Krammer, 1987; van Merriënboer & Paas, 1990). However, for an example to be effective, an “investment of effort” (p. 283) from learners is required and this can be assisted by providing learners with partial example solutions that have to be completed (van Merriënboer & Paas, 1990). The technique requires the learner to study the partial code provided in the completion assignment otherwise they cannot correctly solve the task (van Merriënboer, 1990b; van Merriënboer & Paas, 1990). Partial example solutions promote “mindful abstraction” (van Merriënboer & Paas, 1990 p. 279) and support self-explanation (Stark, 1999), in Renkl, Atkinson et al. (2000).

As described in Chapter 3, the Completion strategy and the Structure-emphasising strategy provide reflective learners with more opportunities for *thinking things through*, to use Felder

⁴ Personal communication, 29/7/2009

and Spurlin's (2005) phrase. By contrast, neither format provides an opportunity for active learners to work with the examples dynamically by *trying things out*, again to use Felder Spurlin's (2005) phrase. As a consequence, active learners may become overwhelmed and experience cognitive overload as they are forced into an uncongenial, more reflective style of learning. Note that learners with a low working memory capacity tend to prefer an active style of learning (Graf, Lin et al., 2008). Moreover, instructional design based on cognitive load theory argues for the careful utilisation of working memory capacity to encourage more effective schema construction (Sweller, van Merriënboer et al., 1998).

Using cognitive load theory, it is argued that the Structure-emphasising strategy increases germane cognitive load because learners are guided by plan-focused self-explanation prompts. A similar benefit should be achieved with the Completion strategy. This is because, following the study of a structure-emphasising worked-example, cognitive schemata (i.e. plan) are further strengthened through solving the completion task. In line with the notion of “worked-out examples function as *analogies*” (van Merriënboer & Krammer, 1987 p. 267), the completion task can be solved without much difficulty by mapping the task with existing schemata – thus drastically decreasing extraneous cognitive load and accordingly more cognitive resources could be allocated for germane activities.

4.2.2 Related work underlying the interface design

We adopted a “dissection” method (Kelley & Pohl, 1996) of explained worked-examples as used in programming textbooks, similar to that implemented in WebEx (Brusilovsky, 2001) - a web-based tool for learning from tutor-explained examples in a programming course. The WebEx interface comprises a program example with bullets (white and green) appended to

the left side of each line of the program. A white bullet indicates that no explanation is available for that line; on the other hand when a green bullet is clicked, the interface displays a textual explanation for the chosen line. Instead of providing explanations for each line of the program, as in WebEx, we propose eliciting self-explanation on various instances of plan structures in the example solutions.

SE-COACH (Conati & VanLehn, 2000) is a computer-based self-explanation tutor within the Newtonian physics domain. Like SE-COACH, we used a masking mechanism for the initial presentation of example solutions and to employ self-questioning prompts (see Webb, 1989). In the SE-COACH, grey boxes are used to masked several parts of the example. As learners uncover part of the example, the interface reveals some text or graphics. Learners are prompted to provide an explanation by means of a self-explain button next to the uncovered part. The self-explain button is designed to provoke the learner to provide an explanation by means of self-questioning (e.g. *“this choice is correct because.....”*). In place of grey boxes, we use a collapsible button (as in WebEx) to hide and unhide the plan structures. Our mechanism has four different rationales. First, it draws the learner’s attention to the underlying plan structure so that self-explanation can be promoted. Second, it helps the learner to abstract away the details of the plan structures linked to the example problem. Third, plan names may well serve as cues to retrieve plan schemas for future problem solving. Finally, it encourages “structural awareness” (Quilici & Mayer, 2002 p. 326).

CORT (Code Restructuring Tool) developed by Garner (2007) supports the part-complete solution method (PCSM) of learning programming. The CORT interface consists of two windows, namely a left and a right window. The right window contains the part-complete

program whereas the left window contains lines of code to be used in the part-complete task along with extra lines serving as *distracters* (Garner, 2007). Some of the lines from the program are left missing for learners to complete. To complete the task, learners move lines of code between the windows and rearrange these lines in the right window via arrow buttons on the toolbar provided within the CORT tool. When learners have completed it, they can copy the program and paste it into a program editor and run it. In this respect, CORT gives freedom to learners to solve the completion task. Nevertheless, this may result in learners coming up with several different solutions to the programming problem, hence it may be difficult to provide specific feedback to learners on their particular final program solution. To alleviate this problem, we used a cloze procedure for program completion, similar to that used by Chang, Chiao et al. (2000).

The next sections describe the two types of interface, one supporting the Structure emphasising strategy and the other supporting the Completion strategy.

4.3 Web-based worked-example system: LECSES

4.3.1 The development environment

LECSES was developed based on a client-server architecture (see figure 4.1) and is a web database application that uses MySQL as its database. The database stores learning materials (which can be retrieved upon request), keeps learners' answers to exercises and a log file that records learners' interaction with the application.

LECSES was built using PHP: Hypertext Pre-processor, or simply known as PHP. It is an HTML embedded scripting language (Ullman, 2004) used to create dynamic web application and commonly used as a server-side scripting language to interact with MySQL. Interaction occurs by establishing a connection between the application, i.e. LECSES (via PHP, which resides on a web server) and the MySQL server. Interaction with MySQL is via the standard database language called Structured Query Language (SQL) (Valade, 2004 p. 15). The following diagram depicts how an interaction occurs between the client (i.e. web browser) and the web server that serves the LECSES web pages.

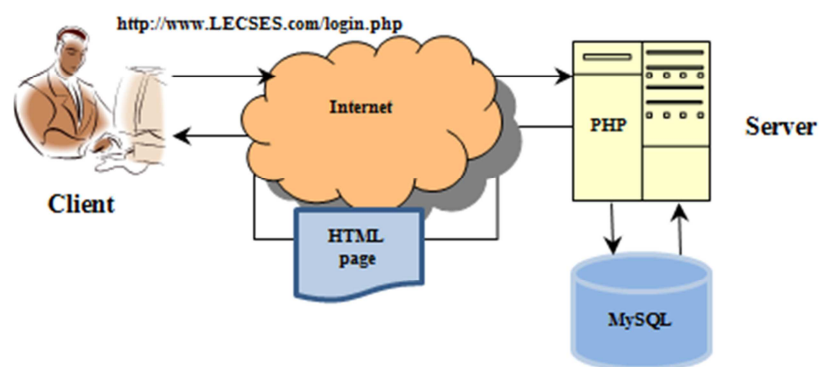


Figure 4.1: Client server architecture

The client sends the request <http://www.LECSES.com/login.php> over the Internet connection to the web server that holds the LECSES web page. The web server reads the PHP script and either processes it or communicates with the MySQL server (where the SQL message is interpreted) and then sends the request back to the client web browser. The web browser displays the HTML page on the screen. Other web development techniques used in developing LECSES are discussed in the subsequent paragraphs.

JavaScript is another scripting language that typically manages activities that occur on the client side (Ullman, 2004). Assisted by jQuery, Javascript was used to extend some special functions of LECSES, such as the collapsible down-pointing triangular button. Clicking on the button unhides / hides the content or a paragraph: in this context, the plan structure (see Figure 4.4).

Style is a set of formatting instructions (i.e. type of font, colour of font, background) that can be used to lay out a web page. Cascading styles means that a set of styles inside a web page override any set of styles included earlier in the web page (Bates, 2006 p. 88). For example, LECSES provides feedback on the correctness or otherwise of an answer for a line of code inserted in a textbox by highlighting its textbox in a different font colour (see Figure 4.13) - a correct answer is highlighted in green and a wrong answer in red.

Asynchronous JavaScript and XML (Ajax)⁵ is a means of transferring data asynchronously between a client and a server without interfering with the content or behaviour of the web page the user is currently viewing. In other words, Ajax provides dynamic client-server interaction without reposting or reloading the web page (Vohra, 2008). For example, as the user clicks on a Hint button, the LECSES interface displays a hint without *reloading* the whole of the web page (see Figure 4.6).

⁵ [http://en.wikipedia.org/wiki/Ajax_\(programming\)](http://en.wikipedia.org/wiki/Ajax_(programming))

The LECSES editor was customised based on TinyMCE⁶, an editor developed by Moxiecode Systems AB (released as Open Source under LGPL). It is a JavaScript-based WYSIWYG (What You See Is What You Get) editor. Customised functions were especially written for the creation of a plan structure, see Figure 4.18. Other features include a Content Management System for creating learning materials.

4.3.2 The interface

The following section presents a series of snapshots of the interface, describing the interactions involved as the learner engages in learning with the Paired-method strategy. Figure 4.2 depicts the interactions. Each example problem consists of two exercises, and so learning with the Paired-method strategy starts with an explanation and then a reflection exercise (the Structure-emphasising strategy), followed by a completion and then a modification exercise (the Completion strategy). In this illustrating example, the time to complete each exercise is specified as 15 minutes. The dashed line indicates the transition from the Structure-emphasising to the Completion strategy. A questionnaire page is presented to the learner after each problem has been studied / solved.

⁶ <http://tinymce.moxiecode.com/>

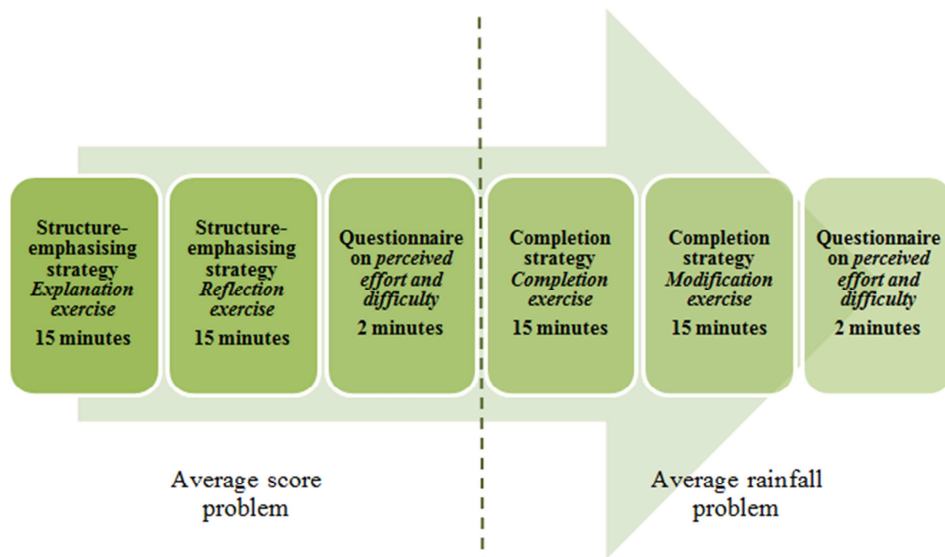


Figure 4.2: The learning interaction (the Paired-method strategy)

4.3.2.1 Structure-emphasising strategy

The first type of interface, supporting the Structure-emphasising strategy is designed to encourage the learner to abstract away the structural details of the example problem and to explain various instances of any plan structures in the example solution.

Explanation exercise

The interface provides a means by which the learner is able to construct a textual explanation, guided by plan-focused prompts. The interface presents the learner with an example problem consisting of a problem description, a sample run, and an example solution (pale blue section, see Figure 4.3) together with a list of programming plan names (see Figure 4.4).

The screenshot shows a web browser window with the address bar displaying a URL from lecses.com. The page content is divided into three main sections: 'Problem', 'Sample run', and 'Solution'.

Problem
Zara is a freelance academic consultant who has been appointed by the principal of Permata Kolej to assess students' personalities based on two test scores, namely Test A and Test B. The personality tests were administered on 20 students. Zara had a program written for her that calculated and displayed the average of the two test scores for each student. Additionally, the program displayed at its end the overall average test score for Test B for female students who scored above 15. This example illustrates a program that prompts the user to input the student's gender and test scores (in the range of 0 to 20) obtained by each student. Assume that the user will enter a valid input.

Sample run:

```
Gender (F for Female / M for Male): M
Score for Test A: 12
Score for Test B: 15
The average of the two test scores was 13.50

Gender (F for Female / M for Male): F
Score for Test A: 12
Score for Test B: 16
The average of the two test scores was 14.00

Gender (F for Female / M for Male): F
Score for Test A: 15
Score for Test B: 17
The average of the two test scores was 16.00
:

Gender (F for Female / M for Male): F
Score for Test A: 10
Score for Test B: 12
The average of the two test scores was 11.00

The average score for Test B (score > 15) for female student was xxx
```

Solution

```
import java.util.Scanner;
public class AverageScore {

    public static void main(String args[]){

        int counter, scoreA, scoreB, female count;
```

The browser's status bar at the bottom indicates 'Internet | Protected Mode: On' and a zoom level of 100%.

Figure 4.3: An example problem consisting of problem description, a sample run, and an example solution (explanation exercise)

The underlying plan structures are initially invisible via a masking mechanism (see Figure 4.4). The exploration of the programming plans is fully under the learner's control. That is, the learner can open several plans at the same time and in no particular order.

Solution

```
import java.util.Scanner;
public class AverageScore {

    public static void main(String args[]){

        int counter, scoreA, scoreB, female_count;
        double total_score, female_scoreB;
        char gender;

        counter = 0;
        total_score = 0.0;
        female_count = 0;
        female_scoreB = 0.0;

        Scanner keyboard = new Scanner(System.in);

        ▼ Loop entry condition, counter-controlled

        ▼ Guard and division plan

    }
}
```

Figure 4.4: Invisible plan structures via the masking mechanism

Clicking on a down-pointing triangular button next to a plan name reveals its structure. A prompt dialog box immediately appears on the right, next to the plan structure being explored, to start an explanation exercise (see Figure 4.5). The prompt dialog box provides a text area for explanation input and a hint button to help the learner with the explanation exercise.

Solution

```
▲ Loop entry condition, counter-controlled
while(counter < 20 ){

    System.out.print("Gender (F for Female / M for Male): ");
    gender = keyboard.next().charAt(0);

    System.out.print("Score for Test A: ");
    scoreA = keyboard.nextInt();

    System.out.print("Score for Test B: ");
    scoreB = keyboard.nextInt();

    total_score = scoreA + scoreB;

    System.out.printf("The average of the two test scores was %.2f " , total_score/2);

    if (gender == 'F' && scoreB > 15){
        ▼ Running total loop plan
    }
    ▼ Counter loop plan
}
▼ Guard and division plan
```

Explain how the expression (counter < 20) in the while statement is evaluated.

Figure 4.5: A prompt dialog box for an explanation input

The interface uses a different font colour for program code. Dark blue denotes various instances of the plan structure in the example solution and red denotes the plan structures currently being explored.

Solution

```
▲ Loop entry condition, counter-controlled
while(counter < 20 ){

    System.out.print("Gender (F for Female / M for Male): ");
    gender = keyboard.next().charAt(0);

    System.out.print("Score for Test A: ");
    scoreA = keyboard.nextInt();

    System.out.print("Score for Test B: ");
    scoreB = keyboard.nextInt();

    total_score = scoreA + scoreB;

    System.out.printf("The average of the two test scores was %.2f ", total_score/2);

    if (gender == 'F' && scoreB > 15){
        ▼ Running total loop plan
    }

    ▲ Counter loop plan
    counter++;
}

▼ Guard and division plan
```

The value of counter at which the loop starts is initialised to 0. Explain the reasoning behind this initialisation taking into account the expression (counter < 20) in the while statement.

Hint 1: Keep track the number of times the user has entered data.

Hint 2: counter will increment itself by 1 each time through the loop.

Submit

Figure 4.6: Hint(s) on demand

Clicking on a Hint button displays Hint 1 and the next click will display Hint 2 (see Figure 4.6). The learner can get one hint or in most cases, two hints. Clicking on an up-pointing triangular button will hide the plan structure. The use of hints to aid explanation is similar to that used in the English Grammar Tutor (Wylie, Koedinger & Mitamura, 2009). The tutor (within the domain of English articles) provides the learner with access to a series of on-demand hints for selecting an article and for explaining that article selection.

Solution

```
System.out.print("Score for Test B: ");
scoreB = keyboard.nextInt();

total_score = scoreA + scoreB;

System.out.printf("The average of the two test scores was %.2f ", total_score/2);

if (gender == 'F' && scoreB > 15){
    ▲ Running total loop plan
    female_count++;
    female_scoreB = female_scoreB + scoreB;
}

    ▲ Counter loop plan
    counter++;
}

▲ Guard and division plan
if (female_count != 0)
    System.out.printf(" The average score for Test B (score > 15) for female student was
    %.2f ", female_scoreB/female_count);
else
    System.out.println(" No average calculated ");

}
```

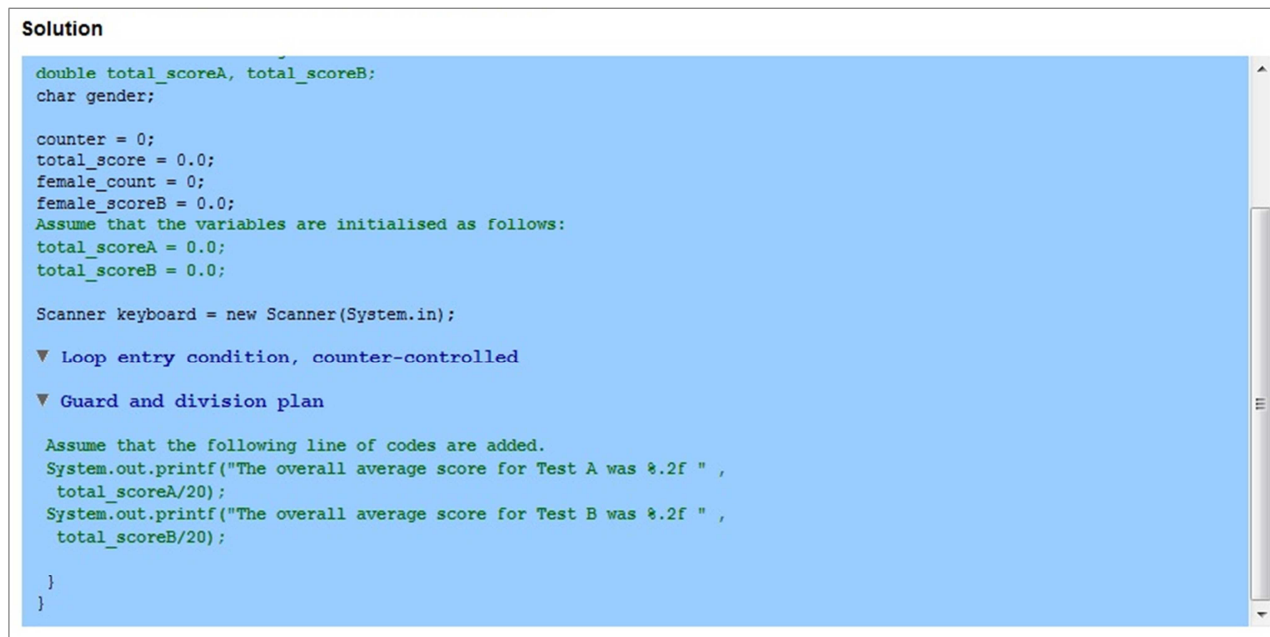
Explain how the running total variables are updated.
The variables are updated if the statement gender == 'F' && scoreB > 15 is TRUE.
Hint 1: Condition under which, the running total variables are updated with the current values.

Figure 4.7: Editing explanation

The learner can edit their explanations by clicking on the edit button (see Figure 4.7). When an edit button is clicked, the text area for explanation input appears. No feedback is given on the learner's explanation. However, after a specified period of time, the interface presents complete descriptions of each of the plan structures to allow the learner to reflect on their own previously generated explanations (i.e. a reflection exercise). The learner works with the explanation exercise within a specified period of time and when the time limit is reached, a response dialog box appears to inform learner to proceed with the reflection exercise.

Reflection exercise

The interface presents complete descriptions of each of the plan structures to allow the learner to reflect on their own previously generated explanations. The underlying plan structures are initially invisible (see Figure 4.8) and plan exploration for the reflection exercise is similar to that of the explanation exercise. Some parts of the program are shown in green (line(s) of code). The learner is required to predict the program's behaviour with respect to these lines and illustrate their answer using an appropriate sample run showing inputs/outputs.



```
Solution

double total_scoreA, total_scoreB;
char gender;

counter = 0;
total_score = 0.0;
female_count = 0;
female_scoreB = 0.0;
Assume that the variables are initialised as follows:
total_scoreA = 0.0;
total_scoreB = 0.0;

Scanner keyboard = new Scanner(System.in);

▼ Loop entry condition, counter-controlled

▼ Guard and division plan

Assume that the following line of codes are added.
System.out.printf("The overall average score for Test A was %.2f " ,
    total_scoreA/20);
System.out.printf("The overall average score for Test B was %.2f " ,
    total_scoreB/20);

}
}
```

Figure 4.8: Predict the program's behaviour (reflection exercise)

Clicking on a down-pointing triangular button next to a plan name reveals the complete description of the plan structure together with the learner's previous answers to the explanation question on the right. The interface also provides a text area at the bottom of an example solution for the learner to type in their answer, as shown in Figure 4.9 (i.e. 9a).

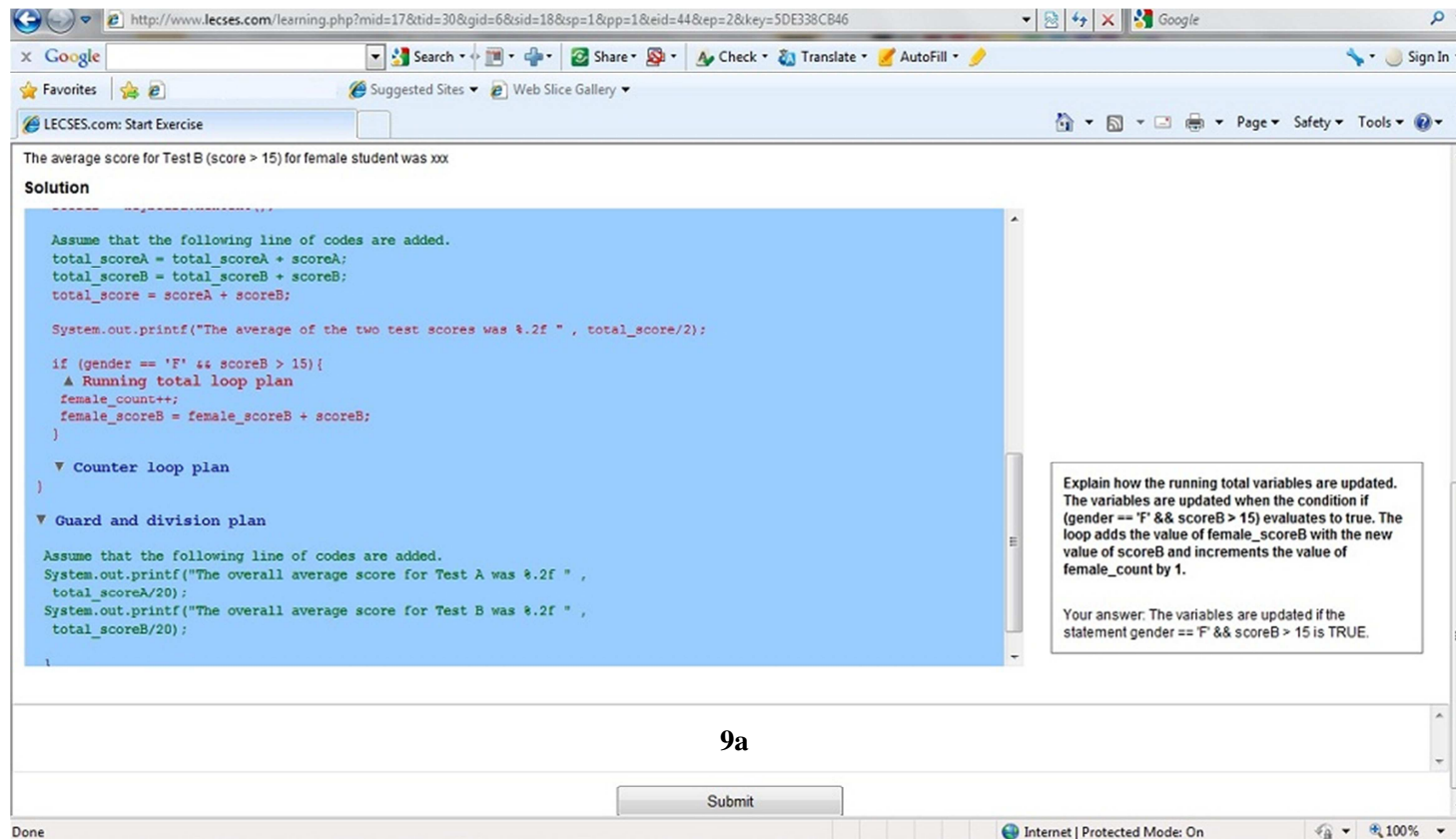


Figure 4.9: Reflection exercise

The learner works with the reflection exercise for a specified period of time and he or she can edit this answer if there is still time.

When the time limit is reached, a response dialog box appears to inform learner to proceed with the questionnaire (see Figure 4.10).

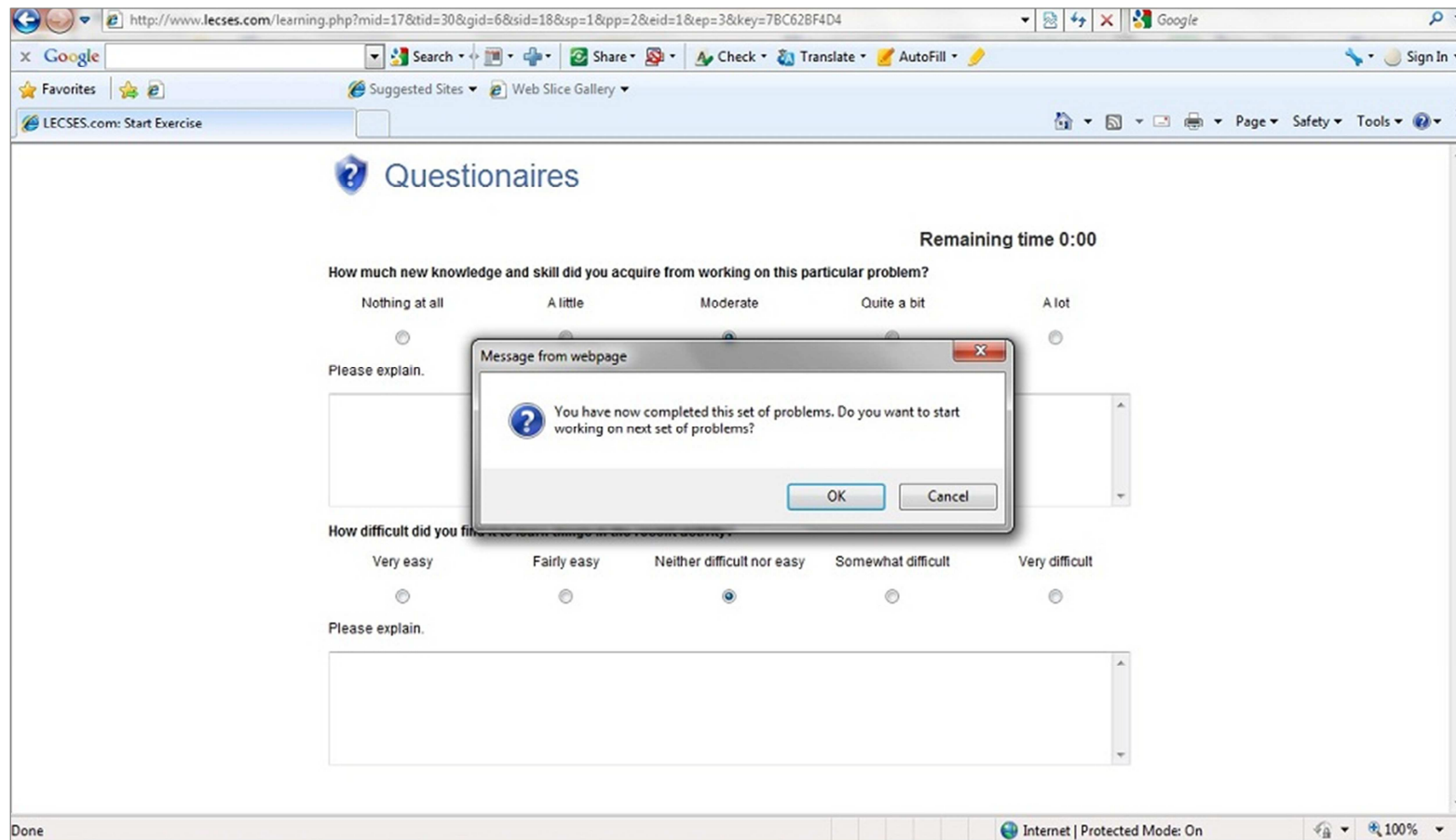


Figure 4.10: A questionnaire page

The questionnaire asks two different questions: (i) “How much new knowledge and skill did you acquire from working on this particular problem?” on a scale ranging from “not at all” to “very much”; (ii) “How difficult did you find it to learn things in the recent activity?” on a scale ranging from “very easy” to “very difficult”. The questionnaire page is presented to the learner **after each example problem solved**. The same questionnaire is used throughout the course. The learner works with the questionnaire for a specified period of time and when the time limit is reached, a response dialog box appears to inform learner to proceed with the completion exercise.

4.3.2.2 Completion strategy

The second type of interface, supporting the Completion strategy, is designed to encourage the learner to complete partial code pertaining to a number of instances of plan structure in the example solution.

Completion exercise

The interface presents the learner with a worked-example consisting of an example problem, a sample run, and a partial example solution (pale blue section, see Figure 4.11) together with a list of programming plan names. The initial presentation of the example solution and code exploration for the Completion strategy is very similar to that of the Structure-emphasising strategy, as already described.

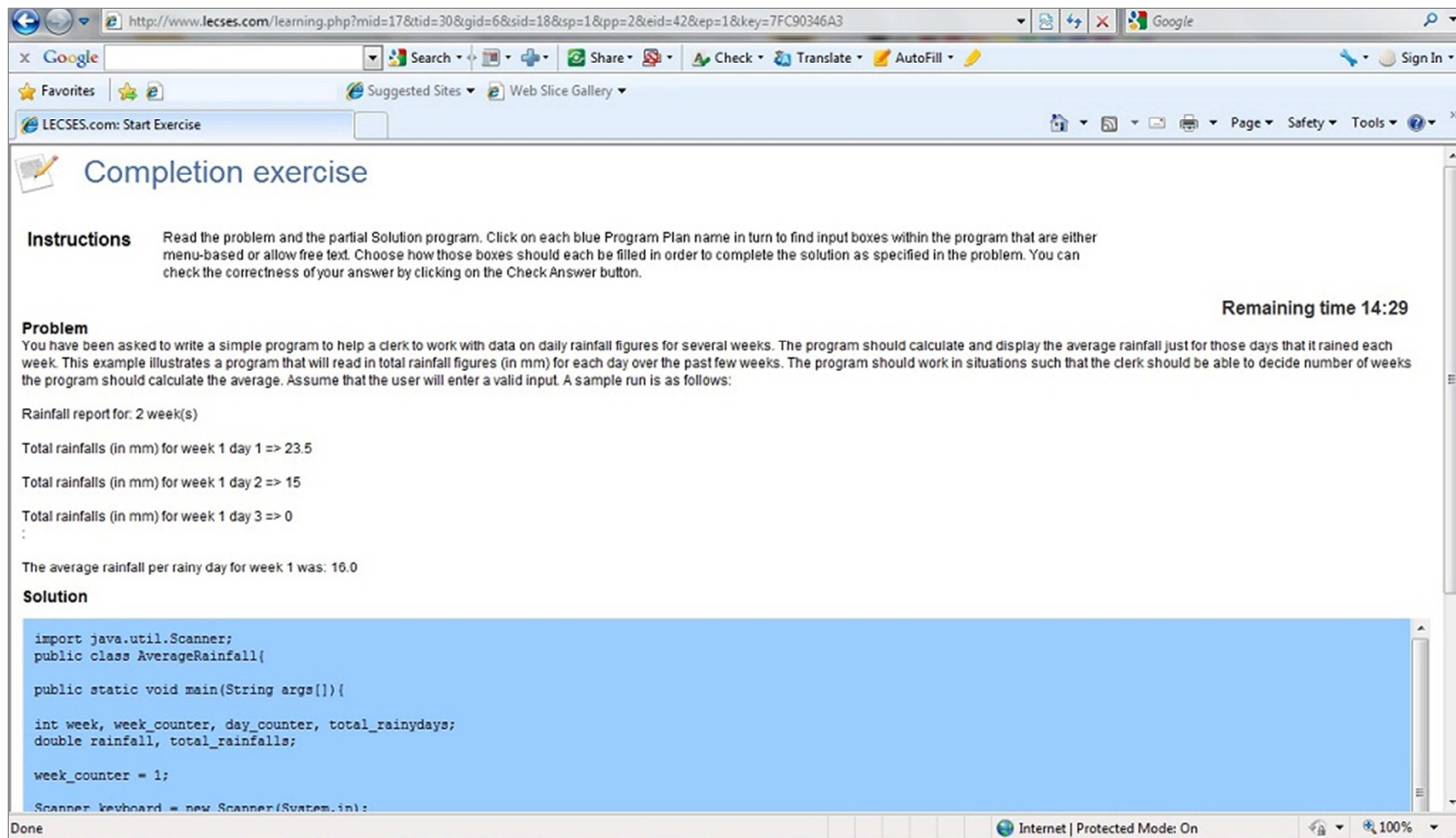


Figure 4.11: An example problem consisting of problem description, a sample run, and an example solution (completion exercise)

Clicking on a down-pointing triangular button next to a plan name (e.g. loop entry condition, counter-controlled plan) reveals the partial code of the plan structure that is either menu-based or allows free text. That is, the interface provides the learner with (i) a

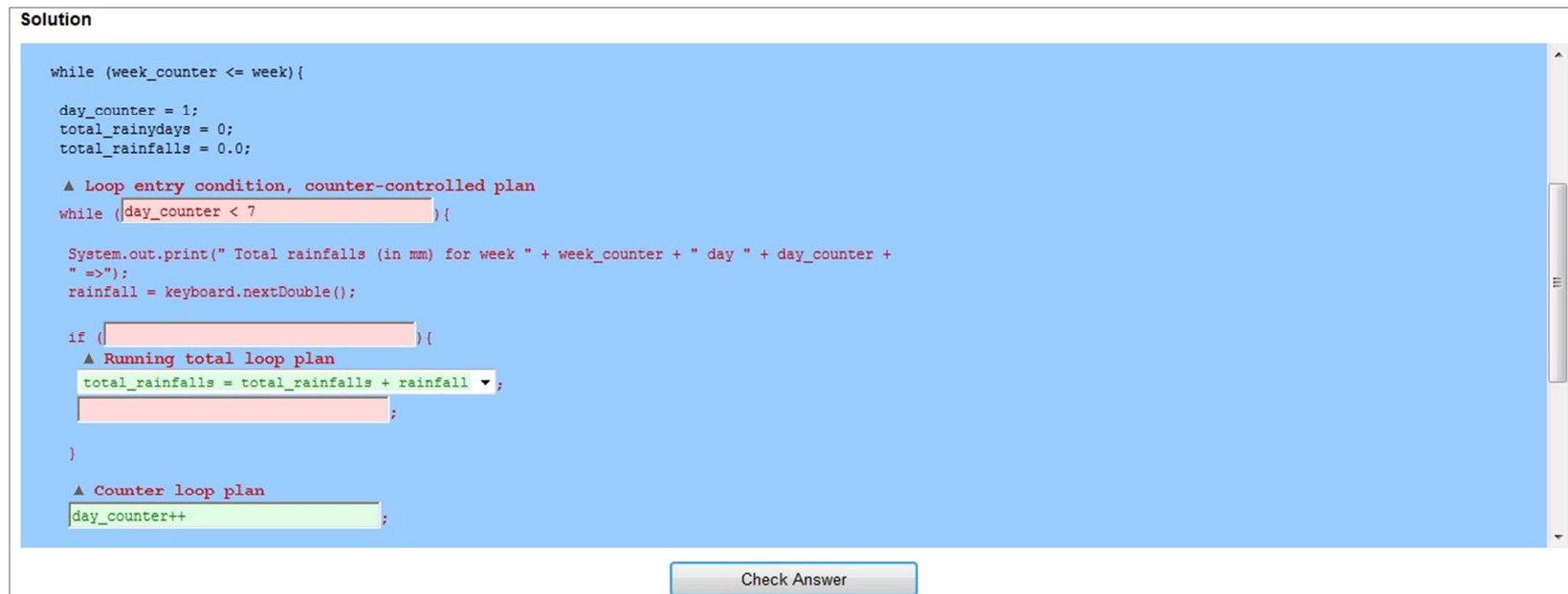
menu to choose a line of code or (ii) a text box in which they have to insert suitable code in order to complete the solution as specified in the example problem (see Figure 4.12).



Figure 4.12: Text box in completion exercise

Note that, certain parts of the example solution (apart from the plan structure) are partially coded and have to be completed by the learner, as shown in Figure 4.12 (i.e. 12a).

When a Check Answer button is clicked, the learner receives immediate feedback on the correctness of answers for the line of code chosen from the menu or the line of code inserted into the textbox (the learner's answer must match the correct answer). A correct answer will be highlighted in green and a wrong answer in red (see Figure 4.13). Also, a dialog box will appear and provide the following response, "Well done" if all answers are correct, otherwise "Some of the completions you have made are not correct". Correcting and re-checking answer is possible provided that the time limit is not reached. When the time limit is reached, a response dialog box appears to inform learner to proceed with the program modification exercise.



Solution

```
while (week_counter <= week){  
    day_counter = 1;  
    total_rainydays = 0;  
    total_rainfalls = 0.0;  
  
    ▲ Loop entry condition, counter-controlled plan  
    while (day_counter < 7){  
  
        System.out.print(" Total rainfalls (in mm) for week " + week_counter + " day " + day_counter +  
            " =>");  
        rainfall = keyboard.nextDouble();  
  
        if ( ){  
            ▲ Running total loop plan  
            total_rainfalls = total_rainfalls + rainfall;  
            ;  
        }  
  
        ▲ Counter loop plan  
        day_counter++;  
    }  
}
```

Check Answer

Figure 4.13: Feedback on the correctness of answer(s)

Modification exercise

The page for the program modification exercise displays the final example solution to a previous completion exercise together with instructions to modify the program. For this exercise, the learner should think about what additions and deletions the program needs to ensure that it solves the modified problem. Exploration of the code is fully under the learner's control – in no particular order.

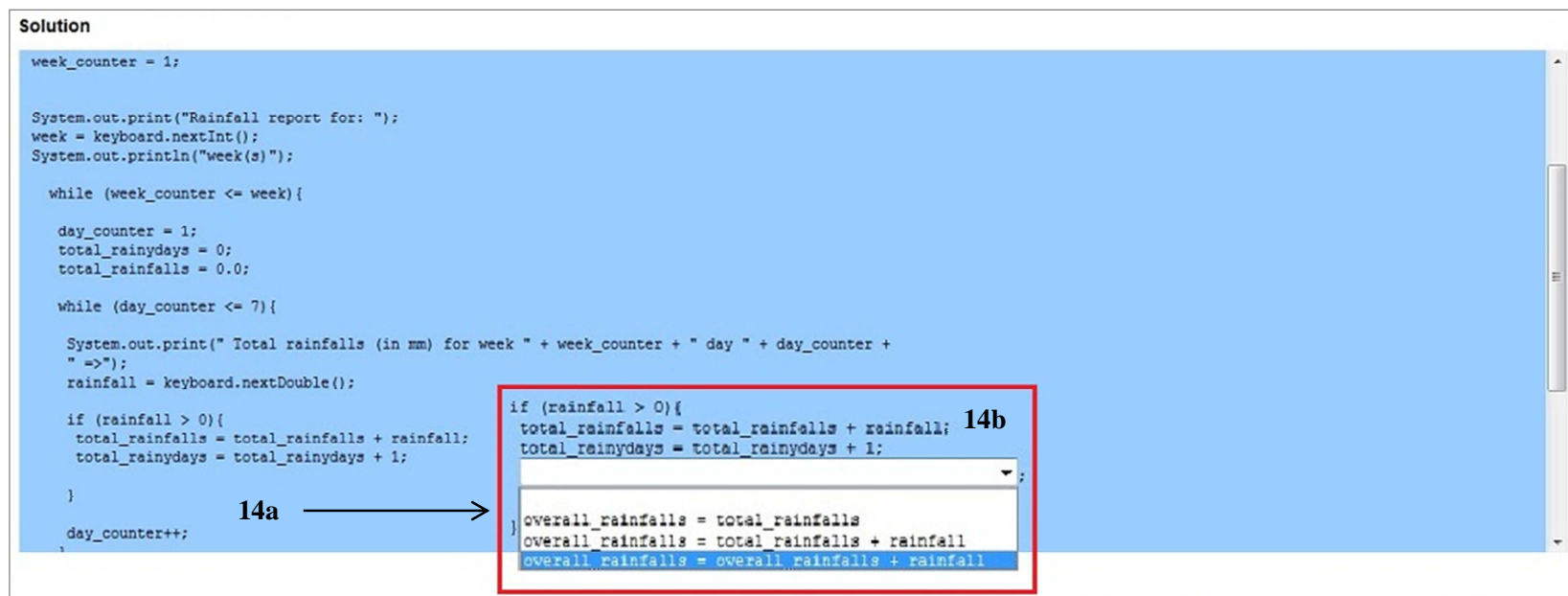


Figure 4.14: Lines of code that are initially invisibly sensitive to being clicked on

Certain parts of the program are **initially invisibly sensitive to being clicked on** - collapsible paragraph(s). That is, when the learner clicks on a non-sensitive part, nothing happens. However, if the learner clicks on a sensitive part of the program see Figure 4.14 (i.e. 14a), to which they think a change should be made, the interface responds by giving one of five possibilities as follows:

- A menu will open and the learner should make a choice from the list of possible lines of code to insert, see Figure 4.14 (i.e. 14b).
- A text box will appear and the learner should insert a line of code. This is a short-answer question that accepts a word or a short phrase (with blank spaces) into a text box.
- Extra code will just appear (serving as a hint) and the learner need take no further action
- A symbol will appear at the end of the line. The learner should click on the symbol if they think that the line should be deleted.
- Nothing will happen, in which case no modification can be made at that place in the program.

When the Submit button is clicked, the learner receives immediate feedback on the correctness of the answers. He or she can make further changes if necessary and if there is still time available. Upon clicking the button, a dialog box appears to provide one of the feedback comments, see Table 4.2. Likewise, a correct answer will be highlighted in green and a wrong answer in red as shown in Figure 4.15.

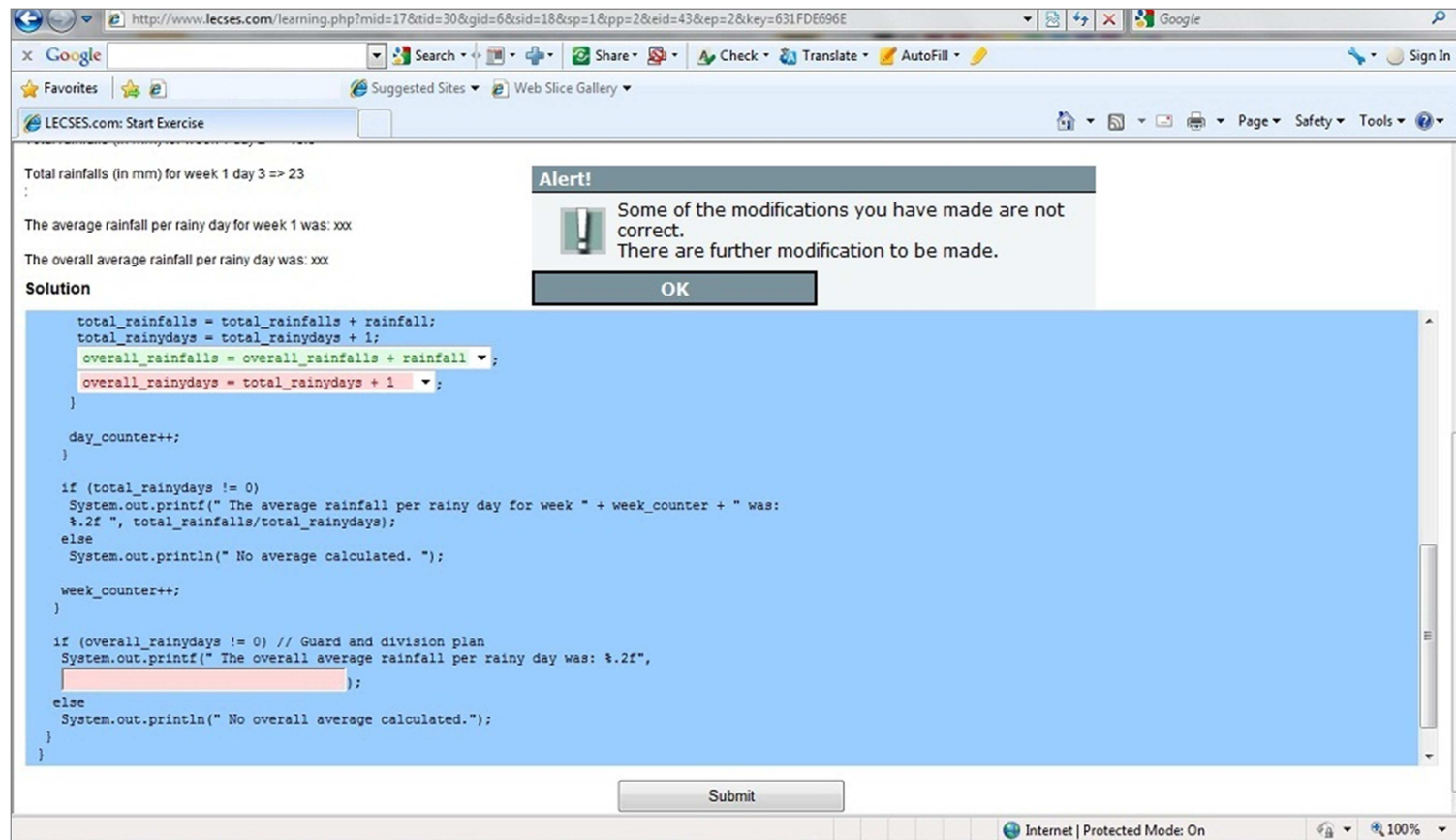


Figure 4.15: Feedback comments dialog box

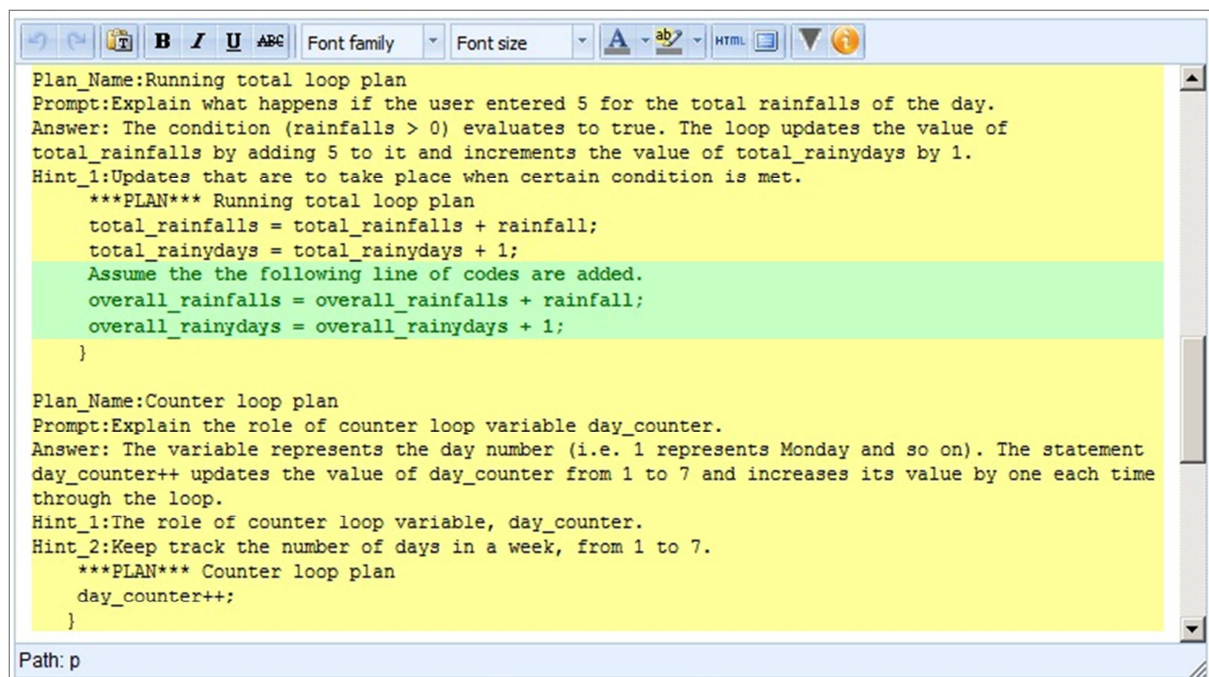
When the time limit is reached, a response dialog box appears in order to inform the learner to proceed with the same questionnaire page as previously encountered. **After working with a set of two example problems**, the learner can choose to discontinue and resume later or proceed with the next set of example problems.

Table 4.2: Feedback comments

	There is no further modification to be made (3)	There are further modifications to be made (4)
All of the modifications you have made are correct (1)	Display “Well done”	Display response 1 and 4
Some of the modifications you have made are not correct (2)	Display response 2 and 3	Display response 2 and 4

4.3.3 The editor

The following section presents the LECSES editor used for composing the example problems.

**Figure 4.16:** Editor for composing the explanation/reflection exercises

Above is a sample screenshot for composing an example problem based on the Structure-emphasising format (see Figure 4.16). The editor allows for the creation of a plan structure, by highlighting a line(s) of code in an example solution, (e.g. `day_counter++;`), in that case the Counter loop plan.

Every plan structure created consists of a meaningful plan name, a prompt to encourage the learner to construct a textual explanation and hints to aid explanation. Note that some lines of code in the program are highlighted in green. These parts are associated with the reflection exercise in which the learner must predict the program behaviour.

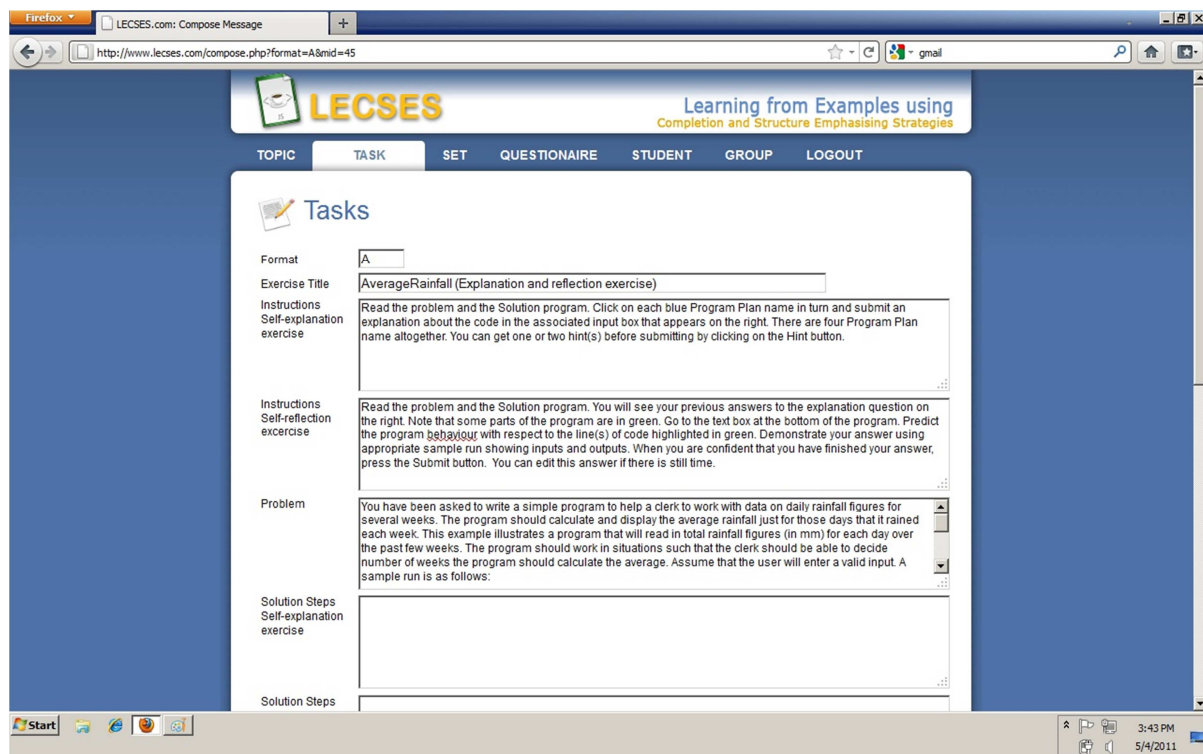


Figure 4.17: Composing an example problem

Figure 4.17 shows a sample screenshot for composing a task (i.e. an example problem) in LECSES. Every task created consists of a title or name of the example problem, instructions for the learner, and a description of the example problem together with its sample run / solution steps.

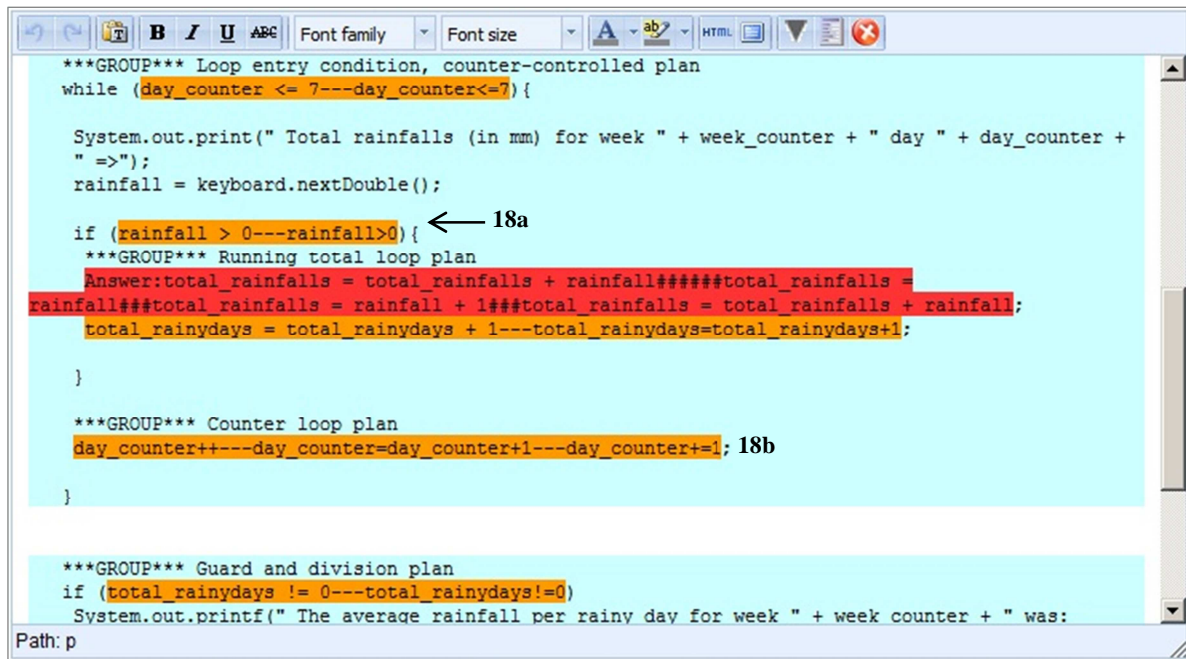


Figure 4.18: Editor for composing a completion exercise

Above is a sample screenshot for composing a completion exercise (see Figure 4.18). The creation of a plan structure is achieved by highlighting a line(s) of code in an example solution, e.g. while (day_counter <= 7), in that case the Loop entry condition, counter-controlled plan. Every plan structure created consists of a meaningful plan name and user-input elements within the example solution that are either menu-based or allow a free text box (highlighted in red and orange, respectively). The menu-based option contains a list of possible lines of code whereas the text box can accept a word or short phrase with blank spaces. That is, the text box can accept several possible correct answers, as long as these answers are listed within the program solution, see 18b. In this case there are three possible answers, separated by “---“. Note that, certain parts of the example solution (apart from the plan structure) are *partially coded* and have to be completed by the learner, as shown in Figure 4.18 (i.e. 18a).

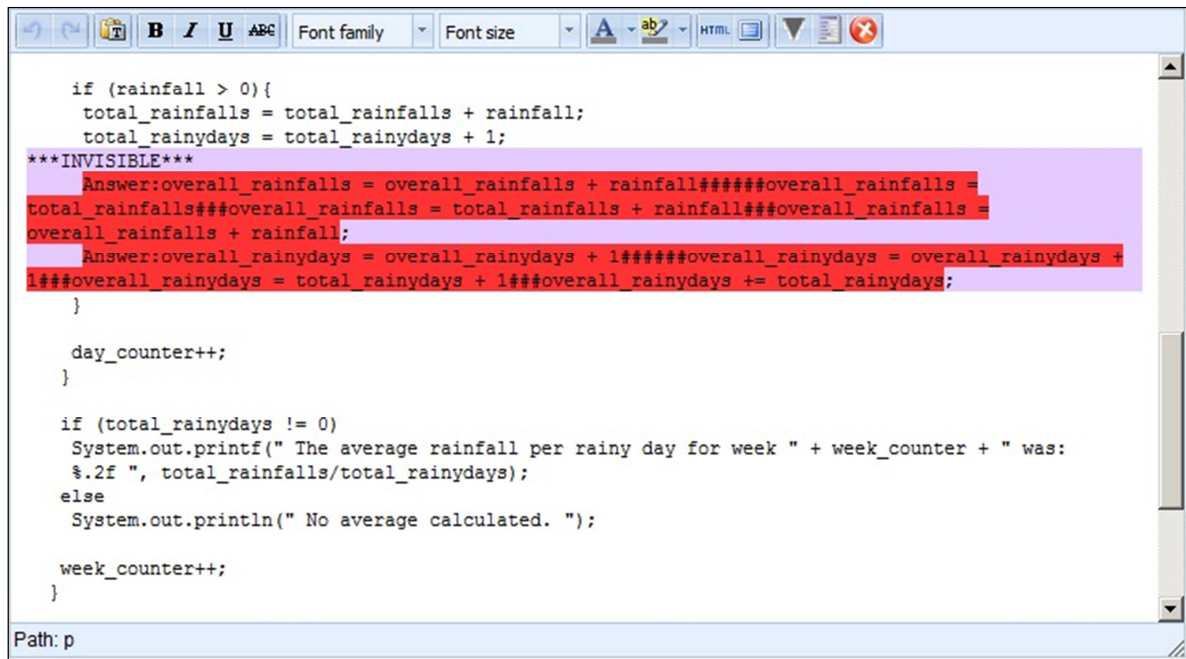


Figure 4.19: Editor for composing a modification exercise

Above is a sample screenshot for composing a modification exercise (see Figure 4.19). Certain lines of code within the program solution are initially invisibly sensitive to being clicked on (purple region). The creation of invisible lines of code is achieved by highlighting these lines and through defining one of the following input elements, a menu or a text box; or by inserting an invisible symbol that will appear at the end of the line (which indicates whether that line should be deleted); or simply inserting extra line(s) of code.

4.3.4 The report generator

The learner's learning activities with LECSES are automatically logged, as depicted in Figure 4.20. Logging occurs each time the learner *interacts* with the system. The pale blue section shows the learner's final answers to an example solution in a completion exercise.

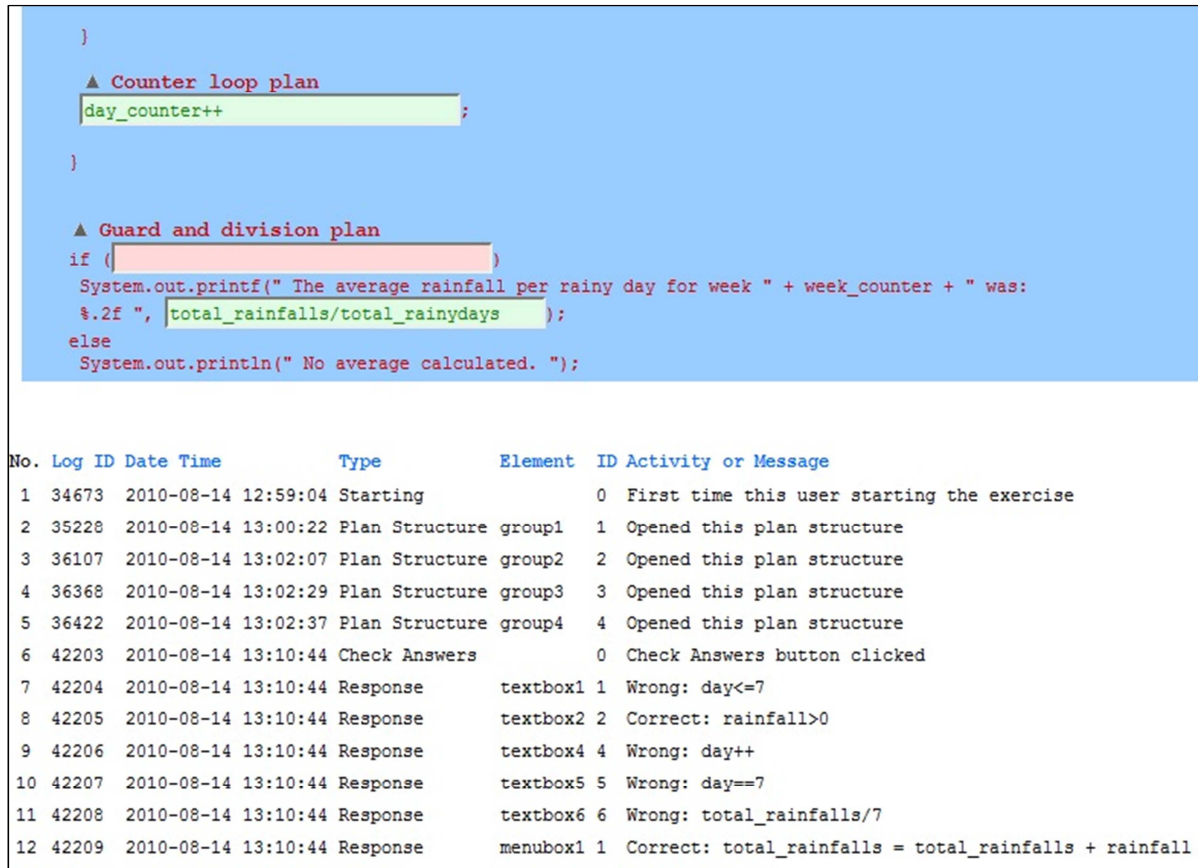


Figure 4.20: LECSES Report generation

4.3.5 The administrator module

Table 4.3 summarises the administrator module for LECSES. The administrator module consists of 5 different sub-modules, from setting up a topic to assigning task(s) to each different group.

Table 4.3: Administrator module for LECSES

	to...
Setting up a topic	<ul style="list-style-type: none"> - publish new topic for the course. - add/edit/remove topic(s).
Creating tasks	<ul style="list-style-type: none"> - add/edit/remove specific instruction. - create a set of 2 example problems. Example problem can be created in two different formats, namely A (structure-emphasising) and B (completion).. - set time on task (time restricted). - Creating task for the three strategies.
Setting up a group / assigning task(s) to each group	<ul style="list-style-type: none"> - create/edit/remove group. - assign learners to one of the 3 groups. - assign tasks to each group.
Questionnaire page	<ul style="list-style-type: none"> - add/edit/remove specific instruction. - create a questionnaire that consists of 2 questions with 5 point rating scales and a text area for an open-ended comment.
Student records	<ul style="list-style-type: none"> - edit/remove record. - insert learners' learning style and working memory capacity to existing records.

4.4 Conclusion

This chapter has presented the design of the Paired-method strategy that aimed to help active learners gain more benefit from being exposed to worked-examples, while at the same time not disadvantaging reflective learners. More specifically, the chapter has discussed the theoretical assumptions and rationale for the design of the Paired-method strategy and then briefly discussed the LECSES development environment. The chapter also described the web-based interface⁷ supporting the two component strategies of the Paired-method, argued for the approach with reference to the research literature, and outlined their specific interface designs. The next chapter discusses the design of an experiment that tested the research hypotheses that underlie the proposed approach.

⁷ LECSES was designed by the author and developed by Mr. Regan Rajan, freelance web-based application developer.

Chapter 5 Experimental design

5.1 Introduction

With reference to the research questions and hypotheses (see Chapter 3), the following objectives for the main experiment were identified. The first objective of the main experiment was to examine any differential effects with respect to the learning *process*. To answer this question, we examined variations of cognitive load (i.e. germane and extraneous cognitive load) that occur during learning, either with the Structure-emphasising strategy or with the Completion strategy or with the Paired-method strategy. The second objective was to investigate any differential effects with respect to the learning *outcomes* in terms of the quality of acquired schemata, including transfer performance, from learning with any of the three worked-example strategies. The third objective was to examine the degree to which individual learning style might influence the learning *process* and *outcomes*. Specifically, the aim was to investigate whether the Paired-method strategy could be employed to equalise the learning *outcomes* on both active and reflective learners. The fourth objective was to investigate the interaction between the active and reflective learning styles and learners' working memory capacity, i.e. the *indirect relationship* identified by Graf, Lin et al. (2008). We chose the topic of loops in the Java programming language as the domain for the main experiment. This chapter is organised as follows. The chapter starts by briefly describing a pilot experiment conducted prior to the main experiment. Next, the chapter discusses the phases and procedures of the main experiment, covering the experimental materials, the instruments, and the participants. The later sections explain the proposed statistical analyses and variables.

5.2 Pilot experiment

A pilot experiment was carried out at the Faculty of Computer Science and Information Technology, University of Malaya from 17th to 18th of June 2010. The pilot experiment was conducted with the intention of highlighting any practical issues related to experimental design, which included but were not limited to the following: to check that the worked-example problems were of appropriate complexity, to ascertain any problems with or effectiveness of the instruments for measuring cognitive load, to determine the time needed for the tasks, and to gauge learners' reactions towards the different worked-example strategies, i.e. the Structure-emphasising and the Completion strategies.

A total of 6 participants who were repeating Programming 1, subject code WXES1114 during the Special Semester (2009/2010) voluntarily agreed to take part in this pilot experiment. The participants were asked to study/solve any four worked-example problems on loops (there were six worked-example problems altogether), and were told that time-on-task was not to be strictly controlled. However, the participants were told to remain within a time frame of about an hour or so. The participants were given answer sheets on which to write their solutions (i.e. pen and paper based) and received one worked-example problem at a time and in random order. Thus, the sequence of worked-example problem and its category (i.e. average problem, vending machine problem, and game-based problem) varied among the participants, see Table 5.1. Upon completion of each problem, participants were asked to give an estimation of the degree to which they had learned from, and the difficulty of, the materials each using a 5-point rating scale, i.e. for assessing extraneous and germane cognitive load, respectively. Table 5.1 provides a summary of the quality of the participants' explanations (i.e. explanation exercise) and total correct answers (i.e. completion exercise) to the problems they worked on.

Table 5.1: The quality of explanations and total correct answers

Not a real name	Worked-example problems (in the order the problems were solved)	Quality of explanation <i>or</i> #total correct answer	GCL	ECL	Solving time (in minutes)
Azri	Average score	All correct but insufficient explanations (partial)	4	2	18
	Coffee machine	5 points	4	3	9
	Coin tossing	4 points	4	4	22
	Photocopy machine	All correct but insufficient explanations (partial)	3	2	8
Asha	Photocopy machine	All correct explanations	5	2	21
	Coffee machine	2 points	4	3	13
	Rock, paper, scissors	All correct but insufficient explanations (partial)	3	4	16
	Coin tossing	4 points	3	3	13
Huda	Average score	Some correct / incorrect explanations	4	3	21
	Average rainfall	4 points	4	4	10
	Rock, paper, scissors	Some correct / incorrect explanations	4	3	14
	Coin tossing	2 points	4	3	10
Farish	Photocopy machine	Some correct / incorrect explanations	4	2	19
	Coffee machine	2 points	3	3	8
	Rock, paper, scissors	Some correct / incorrect explanations	2	4	14
	Coin tossing	3 points	3	4	9
Azlina	Coffee machine	3 points	4	3	12
	Photocopy machine	Some correct / incorrect explanations	4	3	17
	Coin tossing	3 points	4	5	17
	Rock, paper, scissors	Insufficient explanations / did not explain	5	4	20
Farah	Coffee machine	2 points	5	5	34
	Rock, paper, scissors	Incorrect explanations / did not explain	5	4	28

Note: # Total points that could be earned (0-6) for the Completion strategy.

Table 5.2 presents a summary of the mean solving time, and recorded effort as well as reported difficulty for the six worked-example problems.

Table 5.2: Average solving time and reported effort / difficulty (*rounded*)

Problem	[#] Solving time (in minute)	Effort (1-5)	Difficulty (1-5)
* Average score ($n = 2$)	20	4	3
Average rainfall ($n = 1$)	10	4	4
* Photocopy machine ($n = 4$)	16	4	2
Coffee machine ($n = 5$)	15	4	3
Coin tossing ($n = 5$)	14	4	4
* Rock, paper, scissors ($n = 5$)	18	4	4

Note: * Starred worked-example problems were presented using the Structure-emphasising strategy while the others were presented using the Completion strategy. One participant (i.e. Farah, see Table 5.1) could only study/solve two of the worked-example problems within the time frame given. [#] including time it took to study the *word problem statement* (2 minutes or so).

A survey questionnaire was sent out to the participants via electronic mail for further inquiry regarding their overall experience of working with the two worked-example strategies. In particular, the questionnaire covered a question about any difficulty they experienced with the English language as a medium for the pilot experiment, a question about any preference for either of the two worked-example strategies and more importantly, several questions related to the instruments for measuring cognitive load.

In conclusion, even though we only piloted an explanation exercise and a completion exercise using the Structure-emphasising and the Completion strategy, respectively, we still managed to gather useful data on the issues highlighted above. As was determined through the pilot experiment, we decided that time-on-task (including the time it takes to study the *word problem statement*) should be 17 minutes for each exercise. In the end, the slightly shorter time of 15 minutes had to be allocated (this issue is discussed in Chapter 9), but that was still close to the average time taken in this pilot study.

We also assessed the level of difficulty of the materials. It appeared that the sequence of worked-example problems were of appropriate complexity: see Table 5.2, the column labelled ‘difficulty’. Note that the Average Rainfall problem was supposed to be simple but was regarded as complex by one participant (Huda, see Table 5.1): the reason being that the *word problem statement* was difficult to follow. In addition, the level of difficulty for the other materials written in English was regarded as acceptable, thus indicating that the use of English rather than Malay would not hinder their learning. We also identified an issue related to the plan-focused prompts in that some cases were rather confusing. We made several changes and restructured the prompts, making it clearer for participants how to construct a correct explanation. Moreover, there was a strong preference for the Structure-emphasising strategy over the Completion strategy, even though many of the participants constructed correct but insufficient explanations. As a final point, only one participant clearly understood the questions asked which were meant to assess extraneous and germane cognitive load. So the instrument for assessing cognitive load was altered.

5.3 The design of the main experiment

5.3.1 Phases and procedure of the main experiment

This section discusses the activities of the main experiment that took place in three separate phases (i.e. pre-experimental, learning, and transfer phases) as well as post-experimental phase, see Figure 5.1 for further illustration.

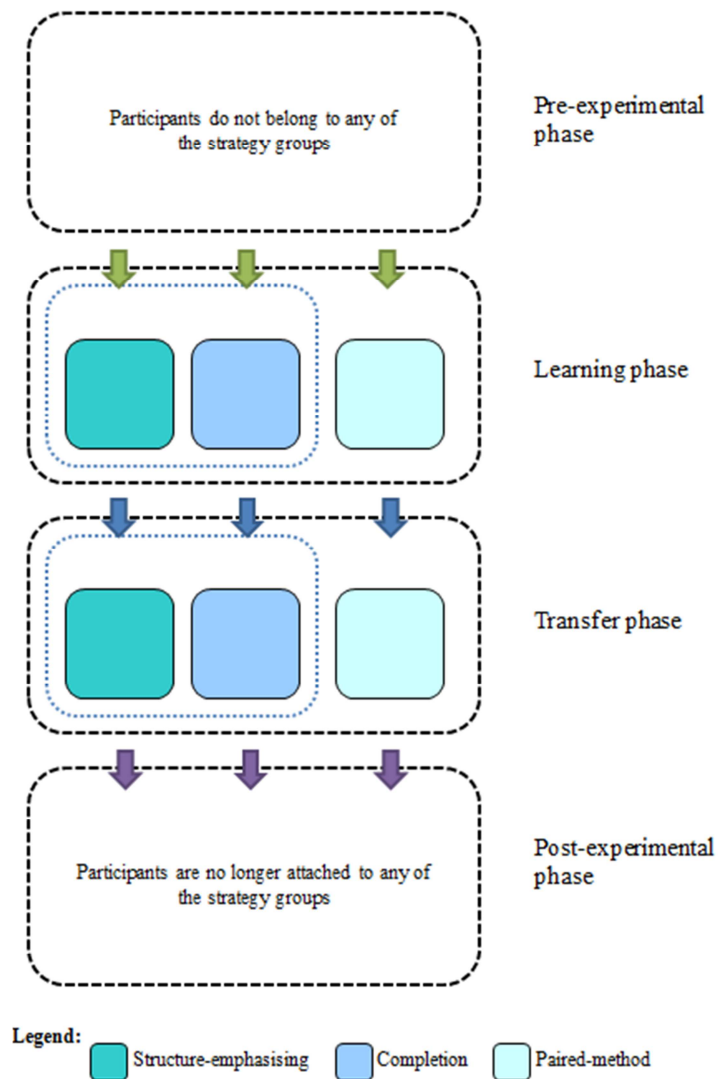


Figure 5.1: Phases and procedure of the experiment

5.3.1.1 Pre-experimental phase

The pre-experimental phase comprised 3 main parts. The first part of the phase started by asking the participants to sign a consent form (see Appendix A) and to complete a questionnaire on their programming background (see Appendix B). The first part of the phase was also concerned with establishing the participants' learning style and working memory capacity. The second part of the phase involved the presentation of the loop theory in Java during the normal lecture session (2 hours).

In addition, the participants were also asked to solve two programming questions on loops during their normal laboratory session (2 hours). Finally, participants were asked to undertake a pre-test to assess their knowledge of a range of basic programming topics including loops. The pre-test will be further discussed in Section 5.3.3. on Instruments.

Specifically, participants were allocated pseudo-randomly (based on learner's learning style) into the three strategy groups - each group had the same number of active, balanced, and reflective learners irrespective of their level of prior knowledge (i.e. programming pre-test scores). Participants were not allocated based on their pre-test scores, the reason being, that the aim of the main experiment was to investigate any differential effects of the strategies on the active and reflective learners.

5.3.1.2 Learning phase

The learning phase started with a briefing session, an introduction to LECSES (a web-based worked-example system), the aims and objectives of the experiment. During the briefing session, participants were introduced to the cognitive load questions concerning their perceived effort and difficulty associated with learning the materials. During the learning phase, the participants were asked to study three pairs of isomorphic worked-example problems, depending on the worked-example strategies, using LECSES (see Appendix G). Each problem consisted of two exercises and a total of 30 minutes (15 minutes per exercise) was allocated for the learning time. Thus the overall learning time was restricted to a maximum of 180 minutes. The activity of learning with LECSES was automatically logged (e.g. the number of times a hint was requested) and time-logged for subsequent analysis. After working with each problem, participants were asked to give an estimate of the difficulty

of the learning method and of the degree to which they had learned new Java concepts from the materials.

5.3.1.3 Transfer phase

The transfer phase started with a briefing session on the procedure and instructions for the transfer test. During the briefing session, participants were introduced to another cognitive load measure, this time concerning an estimate of their mental effort in solving the transfer problem. In particular, this phase was concerned with a program development and coding test and participants were asked to solve 2 near and 2 far transfer problems (4 problems altogether), see Appendix H. After each problem was solved, participants were requested to give an estimate of the mental effort they had invested in solving the problem.

Note that the worked-example problems for the learning phase (and the transfer problems for the transfer phase) as well as the instrument for measuring cognitive load will be further discussed in the section on the LECSES learning environment and experimental variation and in the instruments section, respectively.

5.3.1.4 Post-experimental phase

In the post-experimental phase, participants were asked to complete a questionnaire on their overall experience of taking part in the experiment. The questionnaire largely covered questions on worked-example strategy and LECSES (see Appendix I). Finally, an individual follow-up with a few selected participants who scored in either the Active or Reflective dimension of learning style of the ILS was conducted individually.

5.3.2 LECSES learning environment, experimental materials and variations

5.3.2.1 *Experimental materials*

Figure 5.2 depicts a schematic representation of experimental materials. The letters S, C, and P represent strategy format, i.e. Structure-emphasising, Completion, and Paired-method, respectively.

The experimental materials for the learning phase consisted of 3 sets of two isomorphic worked-example problems, namely Set 1, Set 2 and Set 3. Set 1 consisted of warm-up materials to let learners familiarise themselves with LECSES. Numerals represent the worked-example problem number. Each worked-example consisted of a programming problem, a sample run along with a final program solution to the problem. The worked-example problems in each set were similar with respect to the program's underlying structures (i.e. similar problem category) however each was exemplified by a different surface story. The worked-example problem sets were identical across the strategy groups, but the problems were presented differently according to the strategy format (i.e. Structure-emphasising).

The experimental materials for the transfer phase consisted of 2 sets of two transfer problems. That is, a set of two near transfer problems and a set of two far transfer problems. All the transfer problems were identical across the strategy groups (i.e. strategy-independent). Similarly, the numerals 1-4 represent transfer problems. The following paragraph further describes worked-example problems for the learning phase.

Groups	Warm-up		Learning phase				Transfer phase
	Set 1		Set 2		Set 3		
S	S1	S2	S3	S4	S5	S6	TT1 – TT4
C	C1	C2	C3	C4	C5	C6	
P	S1	C2	S3	C4	S5	C6	

Figure 5.2: Experimental materials

We adopted a within-category example comparison (Gerjets, Scheiter et al., 2008), but with single worked-example presented at a time. The aim was to help learners to extract similarities and differences between the pair of worked-example problems within the same problem category. As previously mentioned, the pair of worked-example problems shared common structural features (i.e. isomorphic plan structures), however they differed with respect to the problem’s surface story. In this way, the aim was for the learners to be able “...to identify features that vary between the category’s examples (i.e., differences) and that are therefore obviously irrelevant with regard to the applicability of the solution principle that is attached to this particular problem category” (Gerjets, Scheiter et al., 2008 p. 77). Furthermore they should have been able to learn that dependence simply on the surface features does not help them in solving a problem (Quilici & Mayer, 2002).

Table 5.3 describes the worked-example problems, arranged in three sets, each involving a different problem category (e.g. Set 1, average problems; Set 2, vending machine problems; and so on). The materials were arranged following a simple-to-complex sequence as proposed by van Merriënboer, Kirschner et al. (2003).

Each worked-example problem comprised four programming plan structures (or three for Set 2). Some programming plans appeared in two different sets (e.g. the counter loop plan) and though these plans served a similar role, their tasks somewhat varied depending on the nature of the problem. To avoid repetitive explanation prompts, we employed different types of prompting question, e.g. to explain the rationale behind an initialisation value of a variable at which the loop starts; to explain how the loop makes its progress; or to explain the condition under which the loop continues / terminates. Figure 5.3 further illustrates this.

Average score	
<i>Prompt:</i>	The value of counter at which the loop starts is initialised to 0. Explain the reasoning behind this initialisation taking into account the expression <code>(counter < 20)</code> in the while statement.
<i>Answer:</i>	That initialisation makes that the expression <code>(counter < 20)</code> in the while statement true before any loop is executed. Counter ensures that each repetition makes progress towards the expression becoming false by increasing its value by one until it reaches 20.
Rock, paper, Scissors	
<i>Prompt:</i>	Explain why is the statement <code>try_count = try_count+1</code> needed within the while loop.
<i>Answer:</i>	The statement ensures that each iteration makes progress toward the condition <code>try_count < MAX_TRIES_ALLOWED</code> becoming false. That is, the statement keeps track of the number of tries the user has spent.

Figure 5.3: Different tactics of prompting questions for the counter loop plan

Table 5.3: Worked-example problems for the learning phase

Average problem		Descriptions	Plan structures
Average score	<i>A program that calculates and displays the average of two test scores for each student as well as the average of one of those tests for female students who scored above 15 points.</i>		Loop entry condition, counter-controlled plan.
	The program additionally calculates and displays at its end the overall average (i.e. class average) test score for the two tests separately.		Running total loop plan.
Average rainfall ¹	<i>A program that calculates and displays the average rainfall just for those days that it rained each week (i.e. weekly average rainfall) for several weeks.</i>		Counter loop plan.
	The program additionally calculates and displays at its end the average rainfall for those days that it rained in the whole period.		Guard and division plan.
Vending machine problem ²		Descriptions	Plan structures
Photocopy machine	<i>A program that prompts the user to make photocopying choices until N is entered (i.e. stop making further choices). The program then computes the cost of the photocopying given the types of document and the number of copies the user has chosen. The program must keep track of the amount of money deposited by the user one coin at a time, calculates and gives back the total change.</i>		Loop entry condition, sentinel-controlled.
	The program allows the user to deposit more money into the machine than needed. When the user wishes to discontinue, ‘-1’ must be entered, however, if the amount of money deposited to this point is not enough, an appropriate message will be displayed.		Running total and limit plan.
Coffee machine	<i>A program that prompts the user to choose a coffee (one choice of coffee per transaction). Coffee will be dispensed when at least the cost of the chosen coffee has been deposited. The program must keep track of the amount of money deposited by the user one coin at a time, calculates and gives back the total change. Finally, the machine should prompt the user to make another selection after a coffee has been purchased.</i>		Valid data entry plan.
	The program allows the user to purchase another selection of coffee with the total change from the previous transaction. The remaining cost of the new coffee must be deposited into the machine before receiving a coffee.		

¹ Problem adapted from Java Gently: Programming Principles Explains by Judy Bishop, 2nd Edition, Addison-Wesley (1998)

² Problem adapted from An Introduction to Programming Using C++ by Kenneth C. Mansfield Jr. and James L. Antonakos, Prentice Hall (1997)

Game-based problem	Descriptions	Plan structures
Coin tossing ³	<p><i>A guessing game program that lets the user plays with the computer-simulated coin-tossing. A game consists of a maximum of seven coin tosses. To win the game, the user must make five correct guesses in that seven, but not necessarily in a row. Finally, the program displays at its end a message depending on whether the user won (a message indicating how many rounds the user took to win the game) or lost the game (a message indicating how many correct guesses the user made prior to losing the game).</i></p> <p>The program lets the user make at least two correct guesses in a row in order to win the game. There is now no maximum of seven coin tosses, but note that the game terminates the first time that the user makes an incorrect guess. Finally, the program displays at its end a message depending on whether the user has won (a message indicating the number of correct guesses in a row) or lost the game (a message indicating “You lost”).</p>	<p>Loop exit condition, counter- and flag-controlled plan.</p> <p>Running total loop plan.</p>
Rock, paper, scissors ⁴	<p><i>A hand-sign game program that lets the user plays with the computer-generated hand-sign. The maximum number of tries allowed is five. The game terminates if the user wins a round before the five tries are used up. When the user chooses a hand-sign, the program replies with a message depending on whether the game is tied, or the user won or lost. Finally, the program displays at its end a message indicating the number of tries it took for the user to win the game or a message indicating the user has reached the maximum number of tries along with number of draw(s) and lost(s) the user made.</i></p> <p>The program prompts the user (at the end of every game) to see if they want to have another go or not. Finally, the program displays at its end a message indicating the number of times the user won out of number of games played along with number of draw(s) and lost(s) the user made.</p>	<p>Counter loop plan.</p> <p>Flag reset plan.</p>

Note: The worked-example problem for the first exercise is highlighted in italics. The slightly different problem requirement for the second exercise is shown in bold, yet is the same underlying problem as the first exercise. See experimental variations section for further information on the exercises.

³ Problem adapted from C by Dissection: The Essentials of C Programming by Al Kelly and Ira Pohl, 3rd Edition, Addison-Wesley (1996)

⁴ Problem adapted from Java Gently: Programming Principles Explains by Judy Bishop, 2nd Edition, Addison-Wesley (1998)

5.3.2.2 *Experimental variations*

The experimental variations for the learning phase were realised as follows. In the Structure-emphasising condition, participants received two different exercises, namely an explanation and a reflection exercise. The first exercise required participants to identify the problem's underlying structure and, in particular, to construct a textual explanation (aided with one or two hints) of various instances of the plan structures in the program solution. No immediate feedback was given to the participants' explanation at this point.

Following the first exercise, participants were presented with complete descriptions of each of the plan structure and they are asked to reflect on their previously constructed explanations. In addition, they were presented with a slightly different problem requirement (in bold, see table 5.3), yet involving the same underlying problem as the explanation exercise (in italics). Note that, for the reflection exercise, the participants were not given any clue apart from the line(s) of code highlighted in green in the program solution (see Figure 5.4). This time participants were asked to predict the program's behaviour with respect to the line(s) of code highlighted and to demonstrate their answer using an appropriate sample run showing inputs and outputs.

```

▲ Running total and limit plan
while (payment < amount_due){

    System.out.print("Please insert coin: ");
    coins = keyboard.nextInt();

    switch(coins){
        case 50 : payment += 0.50;
        break;
        case 20 : payment += 0.20;
        break;
        case 10 : payment += 0.10;
        break;
        ▲ Valid data entry plan
        default : System.out.println("This machine accepts 50 cent, 20 cent, and 10 cent;
        coin only.") break;
    }

    System.out.printf("Amount of money deposited: %.2f ", payment);
}
Assume that the following line of code is added.
balance = payment - amount_due;
System.out.printf(" Your coffee is ready. Balance: %.2f ", payment - amount_due);

System.out.print(" Another selection of coffee? (N to stop the program): ");
coffee = keyboard.next().charAt(0);

```

Figure 5.4: Excerpt from a program solution for reflection exercise

In the Completion condition, participants received two different exercises, namely a completion and a modification exercise. The completion exercise required participants to complete the partial program solution pertaining to a number of instances of plan structure. In particular, they had to complete questions concerning the structure of incomplete solutions to the worked-example based on the given problem specification (van Merriënboer & Paas, 1990). The participants received immediate feedback on the correctness of their answers for the line of code chosen from the menu or the line of code inserted into the textbox.

Following the completion exercise, participants were presented with a complete solution to the previous program completion exercise together with instructions to modify the program. For this exercise, participants had to determine what additions and deletions the program needed to ensure it solved the modified problem.

In the Paired-method strategy, participants received a combination of both worked-example strategies, starting with the Structure-emphasising strategy and then the Completion strategy.

After studying or working with each worked-example problem, participants were requested to give an estimation of the difficulty of the learning method and of the degree to which they had learned any new Java concepts from the materials (i.e. effort).

In the transfer phase, a program development and coding test was administered to measure the participants' ability to transfer their knowledge. In this phase, the participants were asked to solve four programming problems for which the transfer distance varied, i.e. distinguishing between near and far transfer problems.

Near-transfer problems were characterised as having worked-example problem' structures similar to those studied in the learning phase but with new surface stories not encountered previously (see Table 5.4).

Far-transfer problems were characterised as structurally different from the worked-example problems encountered in the learning phase, slightly more complex *word problem statements*, and required different problem solving procedures.

The procedure for the transfer phase was as follows. All of the four transfer problems were identical across the three strategy groups. The test was accompanied with an answer booklet in which participants were to write their programs (i.e. pen and paper based). One problem sheet was given at a time and participants only got the next problem sheet when they had completed and handed in the current one. The participants were given up to 30 minutes to

solve each task, though they could hand in the current one much earlier than the time allotted to proceed with the next problem. Thus, the overall time given to complete these problems was 2 hours. Participants were advised not to go back to a previous problem to continue working on its program solution. After working with each problem, the participants were requested to give an estimation of the mental effort they had invested in solving the problem and the time at which they had completed the problem on the problem sheet given.

Note that the experiment compared the effects of using worked-examples in three ways, with all three designed to encourage learners to engage in cognitive processes relevant to schema construction. Moreover, the experiment was designed in such a way that the intrinsic load was constant as much as possible across the three strategy groups (i.e. without altering what needed to be learned). In other words, the element interactivity essential to the materials to be learned was constant across the three groups. Hence, any differential effects of the three strategy groups could be regarded as a function of the different instructional procedures. Note further, that the experiment was not concerned with the relative strength and weaknesses of the three strategies described above as opposed, say, to a program generation strategy, given the latter's negative consequences for schema acquisition (see Sweller, 1988). Finally, although there is some overlap between the three strategies used, the experiment was designed to tease out the subtle differences between them in terms of their ability to promote schema acquisition and transfer.

Table 5.4: Word problems statement for the transfer phase

Near transfer		Descriptions
Vending machine problem ⁵ <i>Bus ticket machine</i>		To write a program that accepts money for the company's self-service ticket machine. The program should compute the price of the bus ticket, given the Zone the user wishes to go to and the type of journey. The price is computed once the user types N as the Zone (i.e. stop making further choices). The ticket will be dispensed when at least the price of the bus ticket has been inserted. The program must keep track of the amount of money deposited by the user one coin at a time, calculates and gives back the total change.
Game-based problem <i>Hi-Lo number guessing game</i> ⁶		To write a program, called Hi-Lo number guessing game that lets the user guesses the computer-generated secret-number in the least number of tries. The secret number is an integer between 1 and 100, inclusive. When the user makes a guess, the program replies with Hi or Lo depending on whether the guess is higher or lower than the secret number.
Far transfer		Descriptions
Finance-related problem <i>Savings account</i> ⁷		To write a program that prompts the user to enter an amount to be deposited for number of months. This should continue until the user types -1 for the amount. The program should calculate and display the total amount in the account at the end of every month. The program should also display at its end the total interest earned since the start (i.e. the first month). The interest rate is fixed at 5% per annum.
Finance-related problem <i>Depreciation</i> ⁸		To write a program to calculate depreciation of an asset worth RM10,000 over 5 years with expected scrap value of RM1000. The program should display a table summarising on depreciation (e.g., depreciation expense, accumulated depreciation, and so on). The annual depreciation rate is 40%. Note that, care is needed so as to prevent the calculated value of depreciation at end of the fifth year falling below the estimated scrap value.

⁵ Problem adapted from An Introduction to Programming Using C++ by Kenneth C. Mansfield Jr. and James L. Antonakos, Prentice Hall (1997)

⁶ Problem taken from An introduction to object-oriented programming with Java by Wu, C. Thomas, 2nd Edition, McGraw-Hill Higher Education (2006)

⁷ Problem adapted from Introduction to Java Programming by Y. Daniel Liang, 8th International Edition, Pearson (2009)

⁸ Source: Wikipedia

5.3.3 Instruments

5.3.3.1 Index of Learning Styles inventory (ILS)

The Felder-Soloman Index of Learning Styles questionnaire (ILS) (Felder & Soloman, n.d.) is an instrument that helps to determine participants' preferred learning style. The instrument was chosen partly because it has been used previously in the programming education literature. Also, it is freely available and can be accessed at <http://www.engr.ncsu.edu/learningstyles/ilsweb.html>. The ILS questionnaire consists of 44 questions and was administered to the participants to assess their learning style preferences on the four dimensions (i.e. active/reflective, sensing/intuitive, visual/verbal, and sequential/global). Each dimension has 11 questions, with two options for answers related to one or the other category of the dimension (Felder & Spurlin, 2005). For example, Figure 5.5 depicts the active/reflective dimension of the ILS, where scores $-/+$ 5 to 11 indicate a moderate to strong preference and scores $+3$ to -3 indicates a balanced learning style.

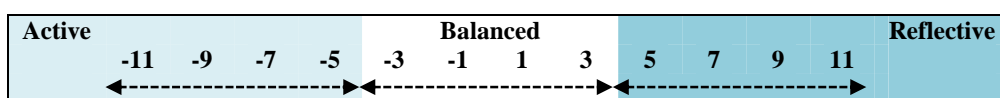


Figure 5.5: The active/reflective dimension of the ILS

The experiment was particularly concerned with the active/reflective dimension of learning styles. Two versions (pen and paper based) of the ILS were administered to the participants. These were an English version and a Malay version, see Appendix C and D. The English version was administered to the International students and that the Malay version was administered to the local students in order to maximise the effectiveness of the instrument.

5.3.3.2 Operation word span (OSPAN)

The operation word span (OSPAN) is a task that helps to determine participants' working memory capacity, as introduced by Turner and Engle (1989). The OSPAN task was administered online using Web-OSPAN, developed by Taiyu Lin and is available at <http://kinshuk.athabascau.ca/webospan/>. We decided to opt for the online version of OSPAN task mainly because it is much easier to administer and monitor a large number of participants at one time. For this task, the participants were shown an arithmetic equation such as $(2 * 5) + 3 = 13$ and they had to answer whether this was true or false. A word was presented after each operation. The equation-word pair was repeated several times (2 to 6 times) and at the end, participants were asked to recall the words in the correct order. At the end of OSPAN task, participants were asked to write down the scores obtained (as the tool could not provide an automatic recording⁹ facility). See Appendix E for the pen and paper based method for recording the scores. As proposed by Turner and Engle (1989), the three OSPAN scores include a process measure, a working memory capacity value (WMC value), and a set size memory span. Whereas the process measure refers to the total number of correct calculations (scores ranging from 0 to 60), the WMC value refers to the total number of correctly recalled words (scores ranging from 0 to 60). The set size memory span refers to the maximum set size of the words recalled correctly (scores ranging from 0 to 6). The latter two measures were also concerned with the correct order of the words recalled.

⁹ Web-OSPAN is built using PHP 4 in which some of the syntax and class used is out dated and no longer supported by the latest PHP version.

5.3.3.3 Programming pre-test

The programming pre-test consisted of two sections. Section 1 covered questions on the declaration and initialization of variables, Boolean operators, and selection statements. Section 2 covered questions related to the topic of loops. The maximum total score was 20 marks. See Appendix F for the programming pre-test.

5.3.3.4 The 5-point rating scale for cognitive load measures

Two different types of cognitive load were assessed during the learning phase, namely germane cognitive (GCL) and extraneous cognitive load (ECL) by adapting the 9-point rating scale developed by Paas (1992) to use a 5-point rating scale. The adapted 5-point rating scale has been used with the aim of assisting the participants to *accurately* evaluate their perceived effort and difficulty. The question for evaluating germane load or effort was “How much new knowledge and skill did you acquire from working on this particular problem?” (the scale ranging from “Not at all” to “Very much”). The question for evaluating extraneous load or difficulty was “How difficult did you find it to learn things in the recent activity?” (the scale ranging from “Very easy” to “Very difficult”), as adapted from Kalyuga, Chandler et al. (1998) and Cierniak, Scheiter et al. (2009). These ratings were logged by LECSES. With regard to the transfer phase, participants were asked to rate their reported mental effort in solving the problem on a 5-point rating scales, ranging from “very low mental effort” to “very high mental effort”. The question for evaluating mental effort was “Please rate your perceived mental effort on solving this problem”. The participants were asked to write down the rating on the problem sheet. The other main source of cognitive load, intrinsic cognitive load (ICL) was not measured. That is, the ICL was kept constant as much as possible across the three strategy groups i.e. by not altering what was needed to be learned across the groups.

5.3.4 Participants

The participants were undergraduate students at the Faculty of Computer Science and Information Technology, University of Malaya and enrolled in an introductory programming course. The bulk of the participants were beginners who had no prior programming experience, however, some participants had a little programming experience.

5.3.5 Scoring

The quality of acquired schemata were determined by a set of criteria (i.e. a marking scheme) applied to the answers to the problems in the transfer tests, as described in Table 5.5. An example of a marking scheme for the Bus Ticket Machine problem is as follows:

Table 5.5: Marking scheme for Bus Ticket Machine problem

Criteria	Points
Logical flow of a program. The following sub-processes must be represented in correct, logical order in a way that makes a program execute its specified task. <ul style="list-style-type: none">- Select a ticket- Make a payment- Calculate total change	1 1 1
Plan structures. Answer must show correct understanding of plan's usage and its role. <ul style="list-style-type: none">- Initialisation plan- Loop entry condition, sentinel-controlled plan- Running total and limit plan- Valid data entry plan	1 1 1 1
Correct usage of selection statement (i.e. if/else or case).	3
Total points that could be earned:	10

Note: 1 points for correct answer or 0 points for incorrect answer.

The selection statement was also counted in the score for it was an essential element of the solution. For the selection statement, a total of 3 points was given for a completely correct usage of the selection statement, however points varied depending on the *accuracy* of answer.

5.4 Proposed statistical analyses and measured variables

The main experiment was largely quantitative. For the empirical evaluation, several measures were taken using subjective ratings (e.g. mental effort), objective measures (e.g. performance post-tests) and instructional efficiency measures (e.g. learning *outcome* efficiency).

Other measures included the Web-OSPAN values, the scores on the ILS instrument that determined preferred learning style and the scores on the programming pre-test, an instrument to measure participants' prior programming knowledge. Analysis of data was undertaken through the use of Statistical Package for Social Science (SPSS).

Prior to doing any statistical analysis, descriptive statistics were performed in order to obtain the mean and standard deviation of each variable. Other tests checked the scale's internal consistency (reliability of a scale) with Cronbach's alpha coefficient. Exploratory analyses of the data for each strategy group were conducted which included a test of normality using the Kolmogorov-Smirnov test and a test of homogeneity of variance using Levene's test. The following sections further describe the statistical analysis, together with the dependent and independent variables.

5.4.1 Web-OSPAN

Pearson's r or Spearman's Rank-order correlation was used to describe the relationship between the WMC values and the process measure (and set size memory span), in terms of the strength of the relationship and its direction (Figure 5.6). Spearman's ρ was used to investigate a relationship between an *ordinal* and a *continuous* variable (Muijs, 2004 p. 155). Figure 5.7 shows the relationship between ILS values and the three Web-OSPAN measures.

		Variable
		<i>Continuous: WMC values</i>
Variables	<i>Continuous: Process measure and set size memory span</i>	

Note: Parametric statistic: Pearson's r, non-parametric alternative: Spearman's rho.

Figure 5.6: Exploring relationships between WMC values and other Web-OSPAN measures

		Independent variable
		<i>Continuous: ILS values</i>
Dependent variables	<i>Continuous: Process measure, set size memory span, WMC values</i>	

Note: Parametric statistic: Pearson's r, non-parametric alternative: Spearman's rho.

Figure 5.7: Exploring relationships between ILS values and Web-OSPAN measures

The Chi-square measure describes the relationship between two *nominal* (i.e. categorical) variables, i.e. different learning style categories and WMC categories, see Figure 5.8. The test also determines whether high WMC learners and low WMC learners were differently represented in the different ILS categories.

		Independent variable
		<i>Nominal: ILScategory^b</i>
Dependant variable	<i>Nominal: WMC category^a</i>	

Note: ^a High WMC (≥ 30), low WMC (< 30). ^b Active (scores from -5 to -11), Balanced (scores from -3 to 3), Reflective (scores from 5 to 11). Non-parametric: chi-square.

Figure 5.8: Exploring relationships between ILS category and WMC category

5.4.2 Learning phase

A correlation analysis was used to describe the relationship between the ILS values and the recorded effort (and reported difficulty) in terms of both the strength of the relationship and its direction, see Figure 5.9.

		Independent variable
		<i>Continuous</i> : ILS values
Dependant variables	<i>Continuous</i> : effort and difficulty scores	

Note: Parametric statistic: Pearson's r , non-parametric alternative: Spearman's ρ .

Figure 5.9: Exploring relationships between ILS values and effort (and difficulty) variables

An independent-samples t-test or Mann-Whitney U test was used to test differences between the two independent strategy groups on a continuous measure, i.e. recorded effort and difficulty. In contrast, a one-way ANOVA or Kruskal-Wallis test was used to describe differences between the three independent strategy groups on the measured variables. See Figure 5.10.

		Independent variable		
		<i>Nominal</i> : strategy group		
		Group S	Group C	Group P
Dependent variables	<i>Continuous</i> : effort and difficulty scores			

Note: Parametric statistic: Independent-samples t-test, One-way ANOVA (three or more groups), non-parametric alternative: Mann-Whitney U test, Kruskal-Wallis test (three or more groups).

Figure 5.10: Exploring differences between strategy groups on effort and difficulty variables

A paired-samples t-test or Wilcoxon Signed Rank test was used for repeated measures when the participants were measured under two different conditions, i.e. Set 2 and 3. See Figure 5.11. More specifically, the test was used in part to ensure that the sets corresponded to the desired levels of difficulty, from simple to complex sequencing as proposed by van Merriënboer, Kirschner et al. (2003).

		Independent variable					
		<i>Nominal: strategy group</i>					
		Group S		Group C		Group P	
		Set 2	Set 3	Set 2	Set 3	Set 2	Set 2
Dependent variables	<i>Continuous: effort and difficulty scores</i>						

Note: Parametric statistic: Paired-samples t-test, non-parametric alternative: Wilcoxon Signed Rank Test.

Figure 5.11: Exploring repeated measures for effort and difficulty variables

An independent-samples t-test or Mann-Whitney U test was used to investigate any differential effects of the three different worked-example strategies on the active learners and the reflective learners, see Figure 5.12.

		Independent variables					
		<i>Nominal: strategy group, ILS category</i>					
		Group S		Group C		Group P	
		Act	Ref	Act	Ref	Act	Ref
Dependent variables	<i>Continuous: effort and difficulty scores</i>						

Note: Act = Active learners, Ref = Reflective learners. Parametric statistic: Independent-samples t-test, non-parametric alternative: Mann-Whitney U test.

Figure 5.12: Exploring between-within differences on effort and difficulty variables

5.4.3 Transfer phase

A correlation analysis was used to describe the relationship between the ILS values and the dependent variables, as shown in the following diagram (see Figure 5.13).

		Independent variable
		<i>Continuous: ILS values</i>
Dependent variables	<i>Ordinal: mental effort</i> ^a <i>Continuous: mental effort</i> ^b , time on tests, post-tests ^c	

Note: ^a mental effort rating score on the post-test. ^b the sum of mental effort scores (distinguishing between near and far transfer tests). ^c the sum of post-test scores (distinguishing between near and far transfer tests). Parametric statistic: Pearson's *r*, non-parametric alternative: Spearman's *rho*.

Figure 5.13: Exploring relationships between ILS values and mental effort/time on tests/post-tests

A correlation analysis was used to describe the relationship between the transfer tests and mental effort (and time on tests), as shown in the following diagram (see Figure 5.14).

		Variable
		<i>Continuous: post-tests</i> ^c
Variables	<i>Ordinal: mental effort</i> ^a <i>Continuous: mental effort</i> ^b , time on tests	

Note: ^a mental effort rating score on a post-test. ^b the sum of mental effort scores (distinguishing between near and far transfer tests). ^c the sum of post-test scores (distinguishing between near and far transfer tests). Parametric statistic: Pearson's *r*, non-parametric alternative: Spearman's *rho*.

Figure 5.14: Exploring relationships between post-tests and mental effort/time on tests

For the analysis of the transfer tests, the same parametric and non-parametric techniques as with the learning phase were used to test differences between independent strategy groups on the measured variables. See Figure 5.15.

		Independent variable		
		Nominal: strategy group		
		Group S	Group C	Group P
Dependent variables	Ordinal: mental effort ^a			
	Continuous: mental effort ^b , time on tests, post-tests ^c			

Note: ^a mental effort rating score on a post-test. ^b the sum of mental effort scores (distinguishing between near and far transfer tests). ^c the sum of post-test scores (distinguishing between near and far transfer tests). Parametric statistic: Independent-samples t-test, One-way ANOVA (three or more groups), non-parametric alternative: Mann-Whitney U test, Kruskal-Wallis test (three or more groups).

Figure 5.15: Exploring differences between strategy groups on mental effort/time on tests/post-tests

The same statistical techniques for the repeated measures as for the learning phase were used to investigate any change in the participants' scores across the two conditions (i.e. near and far transfer tests), see Figure 5.16. More specifically, the test was used in part to find out if any of the strategies showed significant changes in scores, and if so, the extent to which that strategy promoted the transfer of programming problem solving skills.

		Independent variable					
		Nominal: strategy group					
		Group S		Group C		Group P	
		Near	Far	Near	Far	Near	Far
Dependent variables	Continuous: mental effort, post-tests						

Note: Parametric statistic: Paired-samples t-test, non-parametric alternative: Wilcoxon Signed Rank Test.

Figure 5.16: Exploring repeated measures for mental effort and post-tests variables

For the analysis of the results, the same parametric and non-parametric techniques as for the learning phase were used to investigate any differential effects of the three different worked-example strategies on the active learners and the reflective learners. See Figure 5.17.

		Independent variables					
		Nominal: strategy group, ILScategory					
		Group S		Group C		Group P	
		Act	Ref	Act	Ref	Act	Ref
Dependent variables	<i>Continuous</i> [#] : mental effort, time on tests, post-tests						

Note: Act = Active learners, Ref = Reflective learners. [#] distinguishing between near and far transfer tests. Parametric statistic: Independent-samples t-test, non-parametric alternative: Mann-Whitney U test.

Figure 5.17: Exploring between-within differences on mental effort/time on tests/post-tests

A correlation analysis was used to describe the relationship between the programming pre-test scores and the overall post-test scores in terms of both the strength of the relationship and its direction, see Figure 5.18.

		Independent variable
		<i>Continuous</i> : pre-test scores
Dependant variables	<i>Continuous</i> : overall post-test scores	

Note: Parametric statistic: Pearson's r, non-parametric alternative: Spearman's rho.

Figure 5.18: Exploring relationships between pre-test and post-test variables

The same parametric and non-parametric techniques as for the learning phase were used to compare the post-test scores of the high prior knowledge learners and the low prior knowledge learners in each of the three strategy groups. See Figure 5.19.

		Independent variables					
		Nominal: strategy group, Hi/Lo category					
		Group S		Group C		Group P	
		Hi	Lo	Hi	Lo	Hi	Lo
Dependent variables	<i>Continuous</i> : overall post-tests						

Note: Act = Active learners, Ref = Reflective learners. Parametric statistic: Independent-samples t-test, non-parametric alternative: Mann-Whitney U test.

Figure 5.19: Exploring between and within differences on overall post-tests

5.4.4 Learning *outcome* efficiency

Learning *outcome* efficiency was measured using the formula introduced by Paas and van Merriënboer (1993). The formula looks at the mental effort reported by the participants during their post-test and their post-test scores, so deriving learning *outcome* efficiency (*IE*) :

$$\frac{(Z_{\text{post-test}} - Z_{\text{reported mental effort on post-test}})}{\sqrt{2}}$$

First, the raw mental effort scores and post-test scores (i.e. henceforth referred as performance) for each participant were standardised across the three strategy groups, yielding z-scores. Then, the mean mental effort z-scores and mean performance z-scores were combined via the learning *outcome* efficiency formula, giving efficiency means (*IE*).

The means of the mental effort z-scores (*R*) and the performance z-scores (*P*) for each strategy group were plotted in a Cartesian graph using a point $P(R, P)$ coordinate system. The upper left of the graph indicates a relative increase in efficiency whereas the lower right indicates a relative decrease in efficiency. The diagonal line $R = P$ indicates zero efficiency ($IE = 0$). The *IE* is determined as a perpendicular distance from a point in the graph to the line (Paas & van Merriënboer, 1993). Finally, participants' learning *outcome* efficiency scores were analysed using a one-way ANOVA.

5.4.5 Learning *process* efficiency

The formula for computing the learning *process* efficiency is similar to the one used in computing learning *outcome* efficiency. This time, the reported difficulty scores and post-test scores were transformed into z-scores using the grand mean across the three strategy groups.

Specifically, the formula looks at the reported difficulty (R) *during learning* and performance in the post-tests (P), so deriving learning *process* efficiency (LE) as shown below. The mean z-scores for each strategy group were represented using a point P(R, P) coordinate system.

$$\frac{(Z_{\text{post-test}} - Z_{\text{reported difficulty}})}{\sqrt{2}}$$

In the same way as for *outcome* efficiency (IE), the relative *process* efficiency (LE) is determined as a perpendicular distance from a point in the graph to the R = P line.

5.4.6 Task involvement

$$\frac{(Z_{\text{recorded effort}} + Z_{\text{post-test}})}{\sqrt{2}}$$

The above formula is an adapted version of the task involvement formula introduced by Paas, Tuovinen et al. (2005). The adapted formula of the task involvement was computed based on effort (i.e. germane) invested *during learning* and performance in the post-tests, so deriving task involvement (INV), where the Zs were standardised scores. In our study, we used effort invested during learning as opposed to mental effort, i.e. the amount of cognitive resources allocated (Paas, Tuovinen et al., 2005 p. 28). We inclined to use perceived effort as it makes sense to determine relative involvement of learners in certain instructional conditions during the learning *process*.

Note that, the studies conducted by Corbalan, Kester and van Merriënboer (2008; 2009) used effort (i.e. germane) or mental effort invested during training, combined with learning outcomes or transfer test scores to calculate task involvement. Nevertheless, the formula used

to calculate task involvement was somewhat different in their two studies; that in the 2008 study: $(Z_{\text{learning outcomes}} - Z_{\text{germane load}}) / \sqrt{2}$, whereas in the 2009 study: $(Z_{\text{transfer test}} + Z_{\text{mental effort}}) / \sqrt{2}$. According to Paas, Tuovinen et al. (2005), the upper right of the Cartesian graph indicates a relative high task involvement whereas the lower left indicates a relative low task involvement.

5.5 Conclusion

The chapter has briefly discussed a pilot experiment which led to an improvement in the design of the main experiment and its instruments. The chapter has also presented the experimental design, covering the phases and procedures of the main experiment, the experimental materials, the instruments and the participants. Lastly, the chapter explained the statistical analyses for measuring the dependant variables of the main experiment.

Chapter 6 The learning *process*

6.1 Introduction

This chapter describes an experiment on learning programming via worked-examples. It focuses on the learning phase of the experiment. The next chapter focuses on the transfer phase.

The aim of this experiment was to investigate how worked-examples affect learning programming. The experiment also examined the degree to which individual learning style might influence the learning process. In particular, the main objective of this experiment was to observe any differential effects on learning using three different worked-example strategies, taking into account learners who score in either the Active or Reflective dimension of learning style of the Felder-Soloman Index of Learning Style (ILS) questionnaire (Felder & Soloman, n.d.). These strategies are the Structure-emphasising strategy, the Completion strategy, and the Paired-method strategy (that combines both the Structure-emphasising and Completion strategies). The effects are cognitive load (i.e. germane cognitive load, extraneous cognitive load), the quality of cognitive schemata acquired, and transfer performance. The focus of the experiment was on the topic of loops in the Java programming language.

The main experiment was conducted from 14th of July to 30th of August 2010 at the Faculty of Computer Science and Information Technology, University of Malaya. Prior to the main experiment, a pilot study (as discussed in Chapter 5) was carried out at the same faculty from

17th to 18th of June with 6 learners who were repeating the course in the Special Semester (2009/2010). The pilot study was conducted in order to assess whether the worked-examples were of appropriate complexity, to ascertain problems of measuring different kinds of cognitive load and to gauge learners' reaction towards different worked-example strategies.

6.2 Overview of research questions and hypotheses for the learning phase

This section provides an overview of the research questions and hypotheses for the learning phase. More specifically, we addressed the following research questions:

1. Effective design of worked-example strategy does not in itself guarantee positive learning outcomes (see Atkinson & Renkl, 2007).
 - a. To what extent does the design of worked-example strategy foster schema acquisition and transfer?
2. Learning style is a factor in determining whether learners benefit from studying worked-examples.
 - a. Does the benefit from studying worked-examples mediated by individual learning styles?
3. Learning style might interact with learners' cognitive load and will determine the quality of cognitive schemata acquired, hence transfer of learning.

- a. Is there any interaction between learning style and learners' cognitive load?
- b. What difference does the learning via worked-examples strategy make to the quality of the cognitive schemata acquired and to the transfer of programming problem solving skills?

In comparing the learning styles with respect to working memory capacity, we tested the following hypothesis:

1. Learners with high working memory capacity tend to prefer a reflective learning style. On the contrary, learners with low working memory capacity tend to prefer an active learning style (Graf, Lin et al., 2008) – (H3).

In comparing the effects of the strategies, we tested the following hypothesis:

2. Given the same amount of time on task with similar instructional content, it was hypothesised that the Paired-method strategy would lead to better learning¹ than with either the Structure-emphasising strategy or the Completion strategy alone (one-tailed). No prediction was made with regard to the direct comparison between the Structure-emphasising strategy and the Completion strategy (H1).

¹ In terms of the learning *process* (i.e. higher effort and lower difficulty)

In comparing the effects of the learning styles, the following hypotheses² were tested:

3. Active learners would perceive their effort and difficulty as lower and higher respectively, than reflective learners in the Structure-emphasising strategy (H4-A1) and in the Completion strategy (H4-A2).
4. Reflective learners would show no difference with respect to their perceived effort and difficulty in the Structure-emphasising strategy (H4-01) and in the Completion strategy (H4-02) as compared to the Paired-method strategy.
5. Active learners would perceive their effort and difficulty just about or equally high and low, respectively in the Paired-method strategy like reflective learners (H4-03).

Note the hypotheses for the transfer phase will be described in the next chapter.

6.3 Methods

6.3.1 Participants

The experiment involved 117 participants. The majority of the learners were first year undergraduate learners undertaking the WXES1116/WXES1114 course (Programming 1) in Semester 1 (2010/2011). A small number of learners repeating the course in this semester were also participants. The experiment was conducted as part of the course topic on loops and

² The H4 hypothesis in Chapter 3 is broken down into 5 sub hypotheses: H4-A1, H4-A2, H4-01, H4-02, H4-03

the transfer test was administered as the learners' mid semester test (i.e. near transfer tests). Learners were given an extra 5% as part of overall assessment as an incentive for taking part in the experiment. Prior to group allocation, a pre-test was administered to assess the learners' level of knowledge in programming. Learners were allocated equally and pseudo-randomly (based the learner's learning style) into the three strategy groups. That is, each group had the same number of active, balanced, and reflective learners irrespective of level of prior knowledge (i.e. pre-test scores). We did not equally allocate learners based on their pre-test scores, the reason being that the main aim of this experiment was to measure the effects of the strategies on the active and reflective learners. Out of the 117 learners who took part in the experiment; data for just 110 learners who had completed all the tasks required for the main phases of the experiment were finally used for analysis. The participants' programming backgrounds across the three strategy groups -- Structure-emphasising ($n = 37$), Completion ($n = 36$), Paired-method ($n = 37$) -- are summarised in Table 6.1.

Table 6.1: Participants' programming backgrounds

		= Background =		
		<i>Yes</i>	<i>No</i>	<i>Total</i>
Group S	Act	3	6	9
	Ref	1	4	5
	Bal	9	14	23
	<i>Total</i>	13	24	37
Group C	Act	5	4	9
	Ref	3	3	6
	Bal	4	17	21
	<i>Total</i>	12	24	36
Group P	Act	2	6	8
	Ref	0	6	6
	Bal	7	16	23
	<i>Total</i>	9	28	37

Note: Group S = Structure-emphasising, Group C = Completion, Group P = Paired-method; Act = Active, Ref = Reflective, Bal = Balanced. Yes means the participant had some background in programming.

The participants' categorical programming background was roughly equal (i.e. the pattern of numbers with and without programming background) across the three strategy groups.

6.3.2 Materials

6.3.2.1 Assessment of Learning Style

The ILS questionnaire (Felder & Soloman, n.d.) was administered to the participants to determine their preferred learning style. The experiment was particularly concerned with the active/reflective dimensions of learning styles. Two versions (pen and paper based) of the ILS were employed, i.e., the English version was administered to the International students and that the Malay version was administered to the local students.

6.3.2.2 Assessment of working memory capacity

The operation word span (OSPAN) (Turner & Engle, 1989) was administered to the participants using Web-OSPAN, developed by Taiyu Lin. The OSPAN is a task that helps to determine participants' working memory capacity.

6.3.2.3 Assessment of cognitive load

Two different types of cognitive load were assessed in the learning phase, namely germane and extraneous cognitive load using 5-point rating scales. The question for evaluating germane load or *effort* was "How much new knowledge and skill did you acquire from working on this particular problem?" (the scale ranging from 1 Not at all to 5 Very much).

The question for evaluating extraneous load or *difficulty* was “How difficult did you find it to learn things in the recent activity? (the scale ranging from 1 Very easy to 5 Very difficult). With regard to the transfer phase, participants were asked to rate their reported mental effort in solving the problem on a 5-point rating scales, ranging from “very low mental effort” to “very high mental effort”. The question for evaluating mental effort was “Please rate your perceived mental effort on solving this problem”.

6.3.2.4 Programming pre-test

As mentioned in the previous chapter, the instrument for the programming pre-test consisted of two sections. The first section covered questions related to fundamental programming knowledge (e.g. declaration and initialization of variables, Boolean operators, and selection statements). The second section covered questions related to the topic of loops. The pre-test was administered 2 days after the learners were given a 2 hour lecture on the topic of loops (both the lecture and the pre-test were conducted in the same week). The following week, the learners were given two lab questions on the topic of loops to work with during their normal laboratory hours (2 hours) and after a further 3 days (during the same week), the learning phase (the main experiment) was conducted.

6.3.2.5 Worked-example problems

The experimental materials for the learning phase (see Figure 6.1) consisted of 3 sets of two isomorphic worked-examples, namely Set 1, Set 2 and Set 3. Set 1 was treated as warm-up materials to help the learners familiarise themselves with the LECSES system. Letters (S, C, and P) indicate strategy format, (i.e. Structure-emphasising, Completion, and Paired-method)

for Group, respectively. Numerals represent the worked-example problem number. The worked-example problem sets were identical across the strategy groups, but the problems were presented differently according to one of the three strategy formats. The worked-example problems in each set were similar with respect to the program's plan structures (i.e. similar problem category) however each was exemplified by a different surface story. The nature of the worked-example problems were discussed in Chapter 5. Each worked-example consisted of a programming problem, a sample run along with a final program solution to the problem.

Groups	Warm-up		Learning phase				Transfer phase
	Set 1		Set 2		Set 3		
S	S1	S2	S3	S4	S5	S6	TT1 – TT4
C	C1	C2	C3	C4	C5	C6	
P	S1	C2	S3	C4	S5	C6	

Figure 6.1: Experimental materials

Materials for the transfer phase consisted of 2 sets of two near and far transfer problems. These materials will be described in the next chapter.

6.3.3 Method for statistical data analysis

Preliminary analyses were conducted prior to running any of the statistical analysis. The preliminary analysis involved running descriptive statistics on each variable in order to ascertain the characteristics of the sample (for example, mean and standard deviation) as well as to check the variables for violation of the assumptions of normality, linearity, and homoscedasticity (Pallant, 2007 p. 53). The variables were discussed in detail in Chapter 5. Another pre-analysis test was to check each scale's internal consistency, calculated by

Cronbach's alpha coefficient; that is to check the reliability of subjective ratings such as reported mental effort.

The analysis for the learning phase and the transfer phase were conducted separately. The analysis was conducted in two ways, by comparing the effects of the strategies and by comparing the effects of the learning styles. The analysis for comparing the effects of the strategies aimed to observe any **differential effects across the three strategy groups**. By contrast, the aim of the analysis for comparing the effects of the learning styles was to investigate any **differential effects using the three different worked-examples strategies for the active learners and the reflective learners**. The analysis for comparing the effects of the learning styles involved only data representing a preference for active learning style (values smaller than or equal to -5) and reflective learning (values greater than or equal to 5) on the active/reflective dimension of the ILS. Note, that the balanced learning style scores ranges from -3 to 3. Figure 6.2 depicts the active/reflective dimension of the ILS.

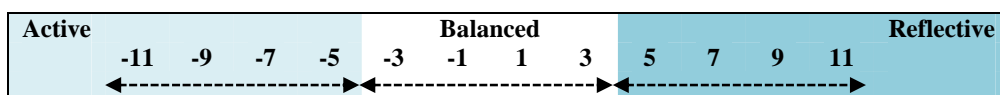


Figure 6.2: The active/reflective dimension of the ILS

In addition, *comparison* analysis methods (see section 6.3.3.2) were performed *between* the different strategy groups as well as *within* each of the strategy group. The subsequent paragraphs discuss the analysis for the Web-OSPAN measures as well as the analysis for comparing the effects of the strategies and learning styles in detail.

6.3.3.1 Analysis for Web-OSPAN measures

An analysis of correlation was performed between the working memory capacity (WMC) values and the other measures of Web-OSPAN (process measure and set size memory span) using Spearman's rho. Note that the distributions for process measure and set size memory span were significantly non-normal. This correlation helped to determine just how significant the WMC values were in relation to working memory capacity (Graf, Liu, Kinshuk, Chen & Yang, 2009). An analysis of correlation was also conducted between the ILS values and all the measures of Web-OSPAN, using Spearman's rho, as the distributions for the ILS data were significantly non-normal. This correlation was computed in order to identify any relationship between the ILS values on the active/reflective dimension and working memory capacity. That is, to see if the result of the correlation analysis was in agreement with the identified *indirect relationship* (Graf, Lin et al., 2008). A follow-up analysis was conducted using the Kruskal-Wallis test if the results indicated a significant correlation.

Additionally, cross tabulation and chi-square tests were used to explore the relationship between pairs of categorical variables. Prior to running any of the tests, the values of WMC were transformed into two categories, namely high WMC and low WMC. Values greater than or equal to 30 indicated high WMC whereas values smaller than 30 indicated low WMC. Also, the values of ILS were divided into three categories (active, balanced, and reflective) following Felder and Spurlin's (2005) recommendations (as cited in Chapter 2). The tests were conducted in order to find out whether high WMC learners and low WMC learners were differently represented in the different ILS categories as well as to check if there was an association between the measured variables.

6.3.3.2 Analysis for comparing the effects of the strategies and learning styles

The analysis for comparing the effects of the strategies consisted of a test **to explore differences** between the three strategy groups. In addition, for each of the strategy groups, a test using a **repeated measures** variable was conducted to investigate any change in learners' scores under the two conditions (i.e. Set 2 and 3), refer to Figure 6.1. Additionally, an analysis of correlation **to explore the relationship** between measured variables was conducted for each of the strategy groups. The analysis of correlation helped to identify the strength and direction of the correlation coefficients among the measured variables. Note that the ILS values were used in this instance.

In the analysis for comparing the effects of the learning styles, the same tests were undertaken (with the exception of test for repeated measures) to explore differences between the two learning styles. This time the ILS categories were explored by running a *comparison* analysis method.

The *comparison* analysis was carried out *between* the different strategy groups and also *within* each of the strategy groups. The former was done by splitting the dataset into two different learning style categories and by selecting a grouping variable called Group. On the other hand, the latter was done by splitting the dataset into three different strategy groups and by choosing a grouping variable named ILScategory. The *between* groups analysis aimed at comparing differences between the strategy groups, focusing on active and reflective learning styles separately. Conversely, the *within* groups analysis explored for differences between the two learning styles in each of the strategy group. Table 6.2 summarises the comparison analysis methods.

Table 6.2: Methods for comparison analysis

	Split dataset	Grouping variable:
<i>between</i> the different groups	* Active Reflective	<i>Group</i> Group S and Group P Group C and Group P
# <i>within</i> each of the groups	Group S Group C Group P	<i>ILScategory</i> Active and Reflective

Note: # splitting the dataset causes the statistical analysis that follows to repeat for each strategy group separately whereas the grouping variable is used to obtain statistical reports on the differences between the two learning styles within each strategy group; vice versa for the between groups analysis. * post-hoc comparison.

The subsequent sections discuss the measured variables and analysis of results for the learning phase. The same sections, specifically for the transfer phase will be discussed in Chapter 7.

6.4 Effort and difficulty scores

Prior to the learning phase analysis, the *total* effort scores for each learner were calculated by adding up the recorded effort scores from the two isomorphic worked-example problems of each set (see Figure 6.3). Similarly, *total* difficulty scores consisted of the addition of the reported difficulty scores from the two worked-example problems of each set. Each *total* was needed to investigate any change in learners' scores across the two conditions (Set 2 and Set 3). Note that the 2 sets were measured at two different points and that the two worked-example problems of each set were very different. That is, each set contained two isomorphic problems of a single type that varied in terms of their complexity and loop structure.

The *overall* difficulty or effort scores were each the sum of the *total* difficulty or effort scores from Set 2 and Set 3. In other words, the *overall* scores consisted of the addition of the scores from all four of the worked-example problems altogether. The *overall* scores were needed to

reflect the accumulated germane and extraneous cognitive load (i.e. effort and difficulty, respectively) experienced during the whole learning phase.

Learning phase			
Set 2		Set 3	
S3	S4	S5	S6
Total effort / difficulty scores of Set 2		Total effort / difficulty scores of Set 3	
Overall effort / difficulty scores			

Figure 6.3: Variables for the learning phase

Note that there were several missing values due to unexpected technical problems associated with the LECSES system (will be discussed in Chapter 9). In calculating, the *total* or *overall* reported difficulty scores, if one of the difficulty scores associated with a learner was missing, the *total* or *overall* value was treated simply as missing (nil) and was not replaced by the *mean* value on the remaining difficulty score(s). The same procedure was undertaken for calculating other measured variables.

An exploratory analysis of the data for each group was conducted which consisted of a test of normality using the Kolmogorov-Smirnov test (or K-S test) and a test of homogeneity of variance using Levene's test. The results of the K-S test (see Table 6.3) for *total* effort/difficulty for the Set 2 and for the Set 3 data (all groups) showed that the distributions appeared to be significantly non-normal. Despite that, the results of Levene's test showed (Table 6.4) that the variances were not significantly different. The results of the K-S test for the *overall* effort data (all groups) showed that the distributions appeared to be normal. For the *overall* difficulty data for Group S, the distributions appeared to be significantly non-normal, whereas for Group C and Group P, the distributions appeared to be normal. The

results of Levene's test showed that the variances representing *overall* data were not significantly different.

Table 6.3: Tests of normality (Learning phase)

Group		Kolmogorov-Smirnov		
		<i>Statistic</i>	<i>df</i>	<i>Sig.</i>
<i>Total effort</i> (Set 2)	Structure-emphasising	.149	36	.043
	Completion	.166	34	.018
	Paired-method	.220	36	.000
<i>Total effort</i> (Set 3)	Structure-emphasising	.192	37	.001
	Completion	.135	34	.118
	Paired-method	.139	29	.157
<i>Total difficulty</i> (Set 2)	Structure-emphasising	.148	36	.044
	Completion	.216	34	.000
	Paired-method	.175	36	.007
<i>Total difficulty</i> (Set 3)	Structure-emphasising	.161	37	.016
	Completion	.165	35	.017
	Paired-method	.206	28	.004
<i>Overall effort</i>	Structure-emphasising	.117	36	.200*
	Completion	.145	32	.083
	Paired-method	.143	28	.149
<i>Overall difficulty</i>	Structure-emphasising	.159	36	.021
	Completion	.126	33	.200*
	Paired-method	.132	27	.200*

Note: * This is a lower bound of the true significance.

Thus, for the learning phase, non-parametric tests were applied (with the exception of the *overall* effort scores), despite the fact that the variance in scores was equal across the three strategy groups. Moreover (specifically, the analysis for comparing the effects of the learning styles), such tests were suitable given that each group consisted of small, unequal numbers of active learners and reflective learners.

Table 6.4: Tests of homogeneity of variance (Learning phase)

	<i>Levene Statistic</i>	<i>df1</i>	<i>df2</i>	<i>Sig.</i>
<i>Total</i> effort (Set 2)	.586	2	103	.558
<i>Total</i> effort (Set 3)	.246	2	97	.783
<i>Total</i> difficulty (Set 2)	.300	2	103	.741
<i>Total</i> difficulty (Set 3)	.190	2	97	.828
<i>Overall</i> effort	.604	2	93	.549
<i>Overall</i> difficulty	.565	2	93	.570

The analysis for comparing the effects of the strategies was undertaken using the Kruskal-Wallis test to explore differences in the *total* effort scores and *total* difficulty scores (for each of the sets, distinguishing between Set 2 and Set 3) across the three different strategy groups. A follow-up analysis was conducted using a non-parametric *post hoc* procedure by means of the Mann-Whitney test if the results of the Kruskal-Wallis test indicated significant differences. Note that, the same set of analyses was repeated for *overall* difficulty scores. As previously mentioned, the data for the *overall* effort scores were normal and a parametric test was used. That is, an analysis was undertaken using a one-way between-groups analysis of covariance (ANCOVA) to compare the three different strategy groups on *overall* effort scores, while controlling for learners' ILS values. Additionally, a one-way between-groups analysis of variance (ANOVA) was conducted to explore the difference in *overall* effort scores between the groups.

In addition, an analysis was undertaken using the Wilcoxon Signed Rank test to find out if any of the strategy groups showed any change in *total* difficulty or *total* effort scores from Set 2 to Set 3 and to ensure that the sets corresponded to the desired level of difficulty, as described in Chapter 5.

Besides, an analysis was performed by correlating the *total* effort scores and the *total* difficulty scores on continuous scales with an independent variable, namely the ILS values. This analysis used Spearman's rho rank-order correlation coefficient (Spearman's rho). Specifically, these tests were used to find out whether there was a relationship between the *total* effort scores or the *total* difficulty scores and ILS values. If the result of the correlation showed a significant association between, for instance, the positive pole of active/reflective dimension of the ILS (see Figure 6.2 for illustration) and the measured variables, a follow-up analysis was conducted to compare a reflective preference to a balanced preference over the measured variables using non-parametric *post hoc* procedure by means of Mann-Whitney test. The comparison was performed in order to find out whether there existed significant differences between the two learning style preferences in terms of effort and difficulty scores. In addition, since the focus of this experiment was on medium to strong active and reflective preferences, the comparison was conducted between these preferences.

In the analysis for comparing the effects of the learning styles, *comparison* analysis methods were employed using the Mann-Whitney U test, once by measuring *total* effort scores and *total* difficulty scores for each of the sets, and once by measuring the *overall* scores for effort and difficulty. Planned comparisons were applied to test the research hypotheses. Table 6.5 presents the hypotheses and planned comparisons.

Table 6.5: Hypotheses and planned comparisons

Hypotheses	Planned comparisons
H4-A1: Active learners would perceive their <i>effort (difficulty)</i> as lower (higher) than reflective learners in the S strategy (one-tailed).	<i>within</i> group: Active compared to reflective and particularly looking at Group S.
H4-A2: Active learners would perceive their <i>effort (difficulty)</i> as lower (higher) than reflective learners in the C strategy (one-tailed).	<i>within</i> group: Active compared to reflective and particularly looking at Group C.
H4-01: Reflective learners would show no difference with respect to their perceived <i>effort</i> and <i>difficulty</i> with the S strategy as compared to the P strategy.	<i>between</i> the different groups: Group S compared to Group P.
H4-02: Reflective learners would show no difference with respect to their perceived <i>effort</i> and <i>difficulty</i> with the C strategy as compared to the P strategy.	<i>between</i> the different groups: Group C compared to Group P.
H4-03: Active learners would perceive their <i>effort (difficulty)</i> just about or equally high (low) with the P strategy like reflective learners (one-tailed).	<i>within</i> group: Active compared to reflective and particularly looking at Group P.

Note: Letters S, C, and P represent strategy format for Group S, Group C and Group P, respectively. S = Structure-emphasising strategy, C = Completion strategy, P = Paired-method strategy that combines both the S and the C strategies.

6.5 Results

An alpha level of .05 was used for all the analyses. A Bonferroni adjustment to the alpha value was applied for the follow-up analysis if any of the results indicated significant differences. That is, the alpha value of .05 was divided by the number of comparisons made, hence giving a more stringent alpha value for determining significance level (Pallant, 2007 p. 228). As an effect size measure for parametric tests, we used the eta squared η^2 ; that is, values $<.06$ indicate a small effect, values in the range between .06 and .13 indicate a medium effect, and values $>.13$ indicate a large effect (Cohen, 1988). For a non-parametric test, we used the effect size r , that is, values of .1 indicate small effect, .3 indicate medium effect, .5 indicate large effect (Cohen, 1988). Significant results with an effect size are presented and discussed. In addition, non-significant results are presented only if they provide at least an

effect size of $|r| \geq .05$ (with the exception of the results of correlation) and discussed if the results provide at least a small to medium effect size. Note that we conducted multiple tests using Spearman's rho for investigating the correlations between several dependant variables. This may have caused a Type 2 error to occur, that is, getting a significant result by chance. To minimise the possibility of reaching such a wrong conclusion, a further analysis using the Mann-Whitney U test was used to follow up the correlation finding by adjusting the alpha value according to the number of comparisons being made.

Conclusions are based on a significant result with an effect size and also on a non-significant result, but indicated by at least a small to medium effect size. In the latter case, the result merely indicates a trend. The following paragraphs provide further guidelines on how we derived conclusions.

For a null hypothesis: (1) The null hypothesis was accepted if there was no significant difference and the effect size was small. (2) We failed to reject a hypothesis if there was no significant difference, but there was at least a small to medium effect size. (3) We rejected the null hypothesis if there was a significant difference and there was at least a small to medium effect size. For an alternative hypothesis: (1) The alternative hypothesis was supported if there was a significant difference and there was at least a small to medium effect size. (2) The alternative hypothesis was also supported if there was a significant difference and there was only a small effect size, however a caveat remained with regard to the effect size. (3) The alternative hypothesis was not supported if there was no significant difference and there was only a small effect size.

6.5.1 Internal validity

Prior programming knowledge was analysed, using a pre-test as an instrument, to identify its possible effect on learning and transfer. Despite quasi-randomisation of the participants to groups, the Kruskal-Wallis test revealed that the pre-test scores differed significantly across the three strategy groups (Group S, $n = 35$; Group C, $n = 36$; Group P, $n = 37$), $\chi^2(2, n = 108) = 9.55, p = .008$. Specifically, there was a significant difference between Group C ($Mdn = 9.00$) and Group P ($Mdn = 6.50$), $U = 374.50, z = -3.23, p = .001$, with a medium effect size ($r = -.38$). Note that the distribution for pre-test scores of Group C was non-normal.

Table 6.6: Descriptive statistics for pre-test scores for the three strategy groups

	<i>Group S</i>			<i>Group C</i>			<i>Group P</i>		
	<i>n</i>	<i>M</i>	<i>SD</i>	<i>N</i>	<i>M</i>	<i>SD</i>	<i>n</i>	<i>M</i>	<i>SD</i>
Pre-test scores	35	7.73	3.25	36	8.72	2.64	37	6.76	2.91

Note: Pre-test scores (0-20)

Note that, the learners scored largely in the first section of the pre-test, i.e. on declarations and initialisation of variables, the use of Boolean operators, and IF/ELSE and CASE structures (10 points altogether). The second section of the pre-test was so difficult that very few learners actually scored in that section, i.e. write out the loop's output and write a simple program containing a loop (10 points). We conclude that the learners across the three strategy groups came to the experiment with basic programming knowledge, but with very little knowledge on the topic of loops. It was unfortunate that there was an apparent floor effect for the pre-test and that there were some differences in elementary programming knowledge between the three groups. However the relatively uniform lack of knowledge of loops across the three groups means that the groups were broadly equivalent as far as the main topic of the worked examples was concerned.

Additional analyses (see Chapter 7) were conducted on the transfer data to investigate any differential effect of the learners' level of prior knowledge (categorised post-hoc, based on their pre-test scores) for the three strategy groups separately.

The reasons for not using pre-test score as a covariate to remove the influence of the pre-test score in the dependent variables were two-fold. First, most of the dependent variables measured had significantly non-normal distributions and that there is no non-parametric alternative to ANCOVA. Second, one of the assumptions of using ANCOVA is that the covariate must be independent from the experimental effect (Field, 2009 p. 397). Given that the three strategy groups significantly differed on the pre-test score, the differential effect of the strategies is to some extent confounded with the effect of the covariate (i.e. pre-test score). In other words, using the pre-test score as a covariate even after log-transformation on the dependant variables would not 'control' or 'balance out' those differences (see Lord, 1967; 1969) in Field (2009 p. 397).

Finally, the 5-point *effort* and *difficulty* rating scales used in our study revealed good reliability via Cronbach's α coefficient = .79 and .75, respectively.

6.5.2 Measures from Web-OSPAN

In the following paragraphs, the results of the analysis of the Web-OSPAN measures are presented. Table 6.7 shows the mean scores and standard deviations on the Web-OSPAN measures for the three learning styles.

Table 6.7: Descriptive statistics for Web-OSPAN measures for the three learning styles

	<i>Active</i>			<i>Balanced</i>			<i>Reflective</i>		
	<i>n</i>	<i>M</i>	<i>SD</i>	<i>n</i>	<i>M</i>	<i>SD</i>	<i>n</i>	<i>M</i>	<i>SD</i>
Process measure	26	55.96	5.72	67	55.51	4.87	17	54.65	3.69
WMC	26	30.12	9.54	67	26.75	11.14	17	26.53	9.95
Set size memory span	26	5.12	0.77	67	4.82	0.97	16	5.13	1.03

The analysis of correlations between the WMC values and other measures of Web-OSPAN, namely the process measure and the set size memory span were calculated using Spearman's rho. Prior to the correlation analysis, a pre-analysis was performed so as to check for any violation of the assumptions of normality, linearity, and homoscedasticity. The results showed high significant correlations ($p < .001$) between WMC values and other Web-OSPAN measures. The process measure ($\rho = .350$) and the set size memory span ($\rho = .811$) indicate strong positive correlations to the WMC values. Table 6.8 presents the results.

Table 6.8: Results from Spearman's rho investigating correlation between WMC values and other measures of Web-OSPAN

Correlation between WMC values and	
Process measure	$\rho = .350, p < .001$
Set size memory span	$\rho = .811, p < .001$

The analysis of correlations (see Table 6.9) calculated using Spearman's rho indicated that no significant association were found between the ILS values and the WMC values ($\rho = -.140, ns$), set size memory span ($\rho = -.058, ns$). Instead, there was a significant but weak negative correlation between the ILS values and the process measure ($\rho = -.225, p = .018$).

On the other hand, further analysis calculated by the Kruskal-Wallis test revealed no significant difference in the values of the process measure between active ($n = 26$), balanced ($n = 67$), and reflective ($n = 17$) learners, $\chi^2(2, n = 110) = 3.18, ns$.

Table 6.9: Results from Spearman's rho investigating correlation between ILS values and Web-OSPAN measures

Correlation between ILS values and	
WMC values	$\rho = -.140, p = .145$
Process measure	$\rho = -.225, p = .018$
Set size memory span	$\rho = -.058, p = .547$

Note: The significant result is highlighted in bold.

Additionally, a chi-square test was conducted to find out if there was an association between high or low WMC categories and the three different learning style categories. The chi-square test revealed no significant association between the ILS categories and the WMC categories, $\chi^2(2, n = 110) = 4.32, ns$, with a small effect size (Cramer's $V = .20$).

Conclusion. The analysis of the results for the Web-OSPAN measures of this study provides no indication of the *indirect relationship* drawn from the literature (Graf, Lin et al., 2008). Moreover, the results provide no evidence for the conclusion found from the recent study conducted by Graf, Liu et al. (2009). This could be due to several factors associated with the data of this study and the method used, as briefly discussed in Chapter 9. Thus, the WMC variable could not be used for further analysis because the relationship between working memory capacity and learning styles could not be replicated. Finally, the alternative (H3) hypothesis, that learners with high working memory capacity (low working memory capacity) tend to prefer a reflective (an active) learning style was not supported.

6.5.3 Comparing the effects of the strategies

This section presents the analysis comparing the effects of the strategies. The section starts with the results of the analysis of correlation and the Mann-Whitney U test, followed by the Kruskal-Wallis test and ANCOVA/ANOVA, and finally the Wilcoxon Signed Rank test. Table 6.10 shows the mean scores and standard deviations of the dependent measures for the three strategy groups.

Table 6.10: Descriptive statistics for effort and difficulty scores (*total* and *overall*) for the three strategy groups

	<i>Group S</i>			<i>Group C</i>			<i>Group P</i>		
	<i>n</i>	<i>M</i>	<i>SD</i>	<i>n</i>	<i>M</i>	<i>SD</i>	<i>n</i>	<i>M</i>	<i>SD</i>
Pre-test scores	35	7.73	3.25	36	8.72	2.64	37	6.76	2.91
<i>Total</i>									
effort scores (Set 2)	36	5.92	1.86	34	5.79	2.07	36	5.22	1.66
difficulty scores (Set 2)	36	7.03	1.83	34	6.41	1.65	36	6.86	1.53
effort scores (Set 3)	37	6.08	2.03	34	6.26	1.80	29	5.21	1.59
difficulty scores (Set 3)	37	7.59	1.71	35	5.69	1.47	29	6.96	1.82
<i>Overall</i>									
effort scores	36	11.89	3.26	32	12.19	3.67	28	10.25	2.78
difficulty scores	36	14.61	3.21	33	12.12	2.75	27	13.96	2.77

Note: Pre-test scores (0-20); effort scores (1-5); difficulty scores (1-5)

6.5.3.1 The analysis of results of total effort/difficulty scores

In the following paragraphs, the results of the analysis of the *total* effort and *total* difficulty scores are presented.

Table 6.11: Results from Spearman's rho correlation and Mann-Whitney U test investigating *total* difficulty scores (Set 2)

<i>Group C</i>	
Correlation <i>between</i>	
ILS values and <i>total</i> difficulty scores (Set 2)	rho = .381, <i>p</i> = .026 (2-tailed)
U-test *	
Act/Bal	<i>U</i> = 28.00, <i>p</i> = .012, <i>r</i> = -.48
Act/Ref	<i>U</i> = 11.00, <i>p</i> = .139, <i>r</i> = -.42 (2-tailed)

Note: Significant results are highlighted in bold. Non-significant results are only presented if they provide at least a small effect size of $|r| \geq .05$. * After the Bonferroni adjustment ($.05/2$), significant value is $< .025$. The *p* value represents 2-tailed significance if the result was not in the expected direction.

The relationship between the ILS values and the *total* difficulty scores (of Set 2) for Group C, measured by Spearman's rho indicates a modest positive correlation, $\text{rho} = .381, p = .026$, (see Table 6.11). This correlation indicates that the more active learners tended to report lower levels of difficulty in studying the worked-example problems from Set 2. Further inspection using the Mann-Whitney U test was used to follow up this finding. A Bonferroni adjustment to the alpha value was applied and therefore all results are reported at the significance level of .025. Since the results of the correlation showed a significant association between the negative pole of the active/reflective dimension and the *total* difficulty scores, the U-test compared an active learning style to a balanced learning style. In addition, a comparison was also conducted between the active and reflective learning styles. It appeared that *total* difficulty scores of learners with an active learning style ($Mdn = 6.00, n = 7$) differed significantly from learners with a balanced learning style ($Mdn = 6.00, n = 21$), $U =$

28.00, $z = -2.53$, $p = .012$, with a medium effect size, $r = -.48$. Active learners had a mean rank of 8.00 while balanced learners had a mean rank of 16.67. On the other hand, *total* difficulty scores of learners with an active learning style did not differ significantly from learners with a reflective learning style ($Mdn = 6.50$, $n = 6$), $U = 11.00$, $z = -1.51$, *ns*. In spite of the fact that the result was non-significant, the effect size was medium ($r = -.42$). Active learners had a mean rank of 5.57 while reflective learners had a mean rank of 8.67.

Conclusion. Active learners in Group C reported lower levels of difficulty than balanced learners in studying worked-example problems from Set 2 using the Completion strategy. In contrast, a medium effect size indicated a trend that reflective learners tended to report higher levels of difficulty than active learners.

Table 6.12: Results from Kruskal-Wallis and Mann-Whitney U test investigating *total* difficulty scores (Set 3)

K-test (Group):	$\chi^2 = 20.68$, $p < .001$
U-test *:	
S/C	$U = 261.00$, $p < .001$, $r = -.52$
S/P	$U = 403.00$, $p = .061$, $r = -.19$
C/P	$U = 289.50$, $p = .005$, $r = -.36$ (2-tailed)

Note: Significant results are highlighted in bold. Non-significant results are only presented if they provide at least a small effect size of $|r| \geq .05$. * After Bonferroni adjustment (.05/3), significant value is $< .017$. The p value represents 2-tailed significance if the result was not in the expected direction.

The results of the Kruskal-Wallis test (see Table 6.12) revealed a statistically significant difference in the *total* difficulty scores (of Set 3) across the three strategy groups (Group S, $n = 37$; Group C, $n = 35$; Group P, $n = 28$), $\chi^2 (2, n = 100) = 20.68$, $p < .001$.

A follow-up analysis using the Mann-Whitney U test was used and a Bonferroni adjustment to the alpha value was applied, hence all the results are reported at the significance level of .017. There was a significant difference in *total* difficulty scores between learners in Group S (*Mdn* = 8.00) and in Group C (*Mdn* = 6.00), $U = 261.00$, $z = -4.41$, $p < .001$, and indicated by a large effect size ($r = -.52$). The significant result of the Mann-Whitney U test also revealed that learners in Group C and in Group P (*Mdn* = 7.00) were different in terms of *total* difficulty scores, $U = 289.50$, $z = -2.84$, $p = .005$, with a medium effect size ($r = -.36$). On the other hand, the *total* difficulty scores of learners in Group S did not differ significantly from those of learners in Group P, $U = 403.00$, $z = -1.55$, *ns*, with a small effect size ($r = -.19$).

Conclusion. For *total* difficulty scores, learners in Group S reported higher levels of difficulty than learners in Group C. Also, learners in Group P reported higher levels of difficulty than their counterparts in Group C.

The result of the Kruskal-Wallis test (see Table 6.13) revealed a statistically significant difference in the *total* effort scores (of Set 3) across the three independent strategy groups (Group S, $n = 37$; Group C, $n = 34$; Group P, $n = 29$), $\chi^2(2, n = 100) = 6.03$, $p = .049$.

Table 6.13: Results from Kruskal-Wallis and Mann-Whitney U test investigating *total* effort scores (Set 3)

K-test (Group):	$\chi^2 = \mathbf{6.03}$, $p = \mathbf{.049}$
U-test *:	
S/C	$U = 595.00$, $p = .690$, $r = -.05$
S/P	$U = 390.50$, $p = .054$, $r = -.24$ (2-tailed)
C/P	$U = 326.50$, $p = .020$, $r = -.29$ (2-tailed)

Note: Significant results are highlighted in bold. Non-significant results are only presented if they provide at least a small effect size of $|r| \geq .05$. * After Bonferroni adjustment (.05/3), significant value is $< .017$. The p value represents 2-tailed significance if the result was not in the expected direction.

Further inspection using the Mann-Whitney U test was used to follow up this finding. A Bonferroni adjustment to the alpha value was applied and therefore all results are reported at the significance level of .017. The results of the U-tests revealed no significant difference in the *total* effort scores (Set 3) of learners in Group S ($Mdn = 6.00$) and in Group C ($Mdn = 6.00$), $U = 595.00$, $z = -.399$, *ns*, with a very small effect size ($r = -.05$). Likewise, no significant difference was found between Group S and Group P ($Mdn = 5.00$), $U = 390.50$, $z = -1.93$, *ns*, $r = -.24$ and between Group C and Group P, $U = 326.50$, $z = -2.33$, *ns*, $r = -.29$.

Conclusion. Learners in Group S and P (and C) recorded just about the same amounts of effort in studying worked-examples using the Structure-emphasising and the Paired-method (and the Completion) strategy, respectively. In contrast, a fairly medium effect size indicated a trend that learners in Group C seemed to record higher amounts of effort in studying worked-examples using the Completion strategy than their learner counterparts in Group P.

6.5.3.2 The analysis of results of overall effort/difficulty scores

In the following paragraphs, the results of the analysis of *overall* effort and *overall* difficulty scores are presented.

Table 6.14: Results from Kruskal-Wallis and Mann-Whitney U test investigating *overall* difficulty scores

K-test (Group):	$\chi^2 = 13.69$, $p = .001$
U-test * :	
S/C	$U = 310.00$, $p = .001$, $r = -.41$
S/P	$U = 394.00$, $p = .099$, $r = -.16$
C/P	$U = 272.00$, $p = .009$, $r = -.34$ (2-tailed)

Note: Significant results are highlighted in bold. Non-significant results are only presented if they provide at least a small effect size of $|r| \geq .05$. * After Bonferroni adjustment (.05/3), significant value is $< .017$. The p value represents 2-tailed significance if the result was not in the expected direction.

The result of the Kruskal-Wallis test (see Table 6.14) revealed a statistically significant difference in the *overall* difficulty scores across the three strategy groups (Group S, $n = 36$; Group C, $n = 33$; Group P, $n = 27$), $\chi^2(2, n = 96) = 13.69, p = .001$. A follow-up analysis using the Mann-Whitney U test was used and all the results are reported at the significance level of .017. There was a significant difference in *overall* difficulty scores between learners in Group S ($Mdn = 15.00$) and in Group C ($Mdn = 12.00$), $U = 310.00, z = -3.43, p = .001$, with a medium effect size ($r = -.41$). The significant result of the Mann-Whitney U test also revealed that learners in Group C and in Group P ($Mdn = 14.00$) were different in terms of *overall* difficulty scores, $U = 272.00, z = -2.60, p = .009$, with a medium effect size ($r = -.34$). On the other hand, *overall* difficulty scores of learners in Group S did not differ significantly from learners in Group P, $U = 394.00, z = -1.29, ns$, with a small effect size ($r = -.16$). Note that, these results showed the same results of the U-test involving data representing *total* difficulty scores (of Set 3).

Conclusion. For *overall* difficulty scores, learners in Group S reported higher levels of difficulty than learners in Group C. Likewise, learners in Group P reported higher levels of difficulty than their counterparts in Group C.

A one-way between-groups analysis of covariance (ANCOVA) was conducted to compare the three different strategy groups on their *overall* effort scores. The independent variable was the strategy group (Group S, C, and P) and the dependent variable consisted of the *overall* effort score. Learners' ILS values were used as the covariate. There was a marginally significant difference at $p = .057$ level in *overall* effort scores between the three strategy groups, after controlling for the ILS values. Despite this, the effect size calculated using

partial eta squared indicated a medium effect size (partial eta squared = .06). As a final point, there was no significant relationship between ILS values and *overall* effort scores, after controlling for the independent variable (group).

A one-way between-groups analysis of variance (ANOVA) was conducted to explore the difference in *overall* effort scores between the three different strategy groups. There was a marginally significant difference in *overall* effort scores for the three strategy groups, $F(2, 93) = 3.00, p = .055$. Despite reaching marginally statistical significance, the effect size was medium (eta squared = .06). Note that, this results showed the same result of the K-test involving data representing *total* effort scores (of Set 3).

The post-hoc comparison indicated that the mean score for Group S ($M = 11.89, SD = 3.26$) was no different from Group P ($M = 10.25, SD = 2.78$). Similarly, there was no significant difference in the mean score between Group C ($M = 12.19, SD = 3.67$) and Group P. Likewise, Group S did not differ significantly from Group C. Note that, these results showed the same trend as the results of the U-test involving data representing *total* effort scores (of Set 3).

Conclusion. The alternative (H1) hypothesis, that the Paired-method strategy would lead to better learning than with either the Structure-emphasising strategy or the Completion strategy alone was not supported.

6.5.3.3 The analysis of results for repeated measures for effort/difficulty scores

The following paragraphs describe the analysis, using the Wilcoxon Signed Rank test, to measure any change in the scores on the effort scale and on the difficulty scale with worked-example problems from Set 2 to Set 3, see Table 6.15.

Table 6.15: Results from Wilcoxon Signed Ranked test investigating change in scores on the effort/difficulty scale across two sets of worked-example problem (i.e. Set 2 and 3).

	Group S	Group C	Group P
W-test:			
Total effort		$z = -2.00, p = .045, r = -.25$	
Total difficulty	$z = -2.11, p = .035, r = -.25$	$z = -2.15, p = .032, r = -.27$	$z = -.952, p = .341, r = -.13$

Note: Significant results are highlighted in bold. Non-significant results are only presented if they provide at least a small effect size of $|r| \geq .05$.

The W-test result indicated a statistically significant increase in *total* effort scores, $z = -2.00$, $p = .045$, $r = -.25$ from problem Set 2 ($Mdn = 6.00$, $n = 31$) to problem Set 3 ($Mdn = 6.00$) for Group C. By contrast, *total* effort scores did not change significantly for Groups S and P.

Similarly, the W-test was used to measure any change in scores on the difficulty scale with worked-example problems from Set 2 to Set 3. The result indicated a statistically significant increase in *total* difficulty scores, $z = -2.11$, $p = .035$, $r = -.25$ from problem Set 2 ($Mdn = 7.00$, $n = 36$) to problem Set 3 ($Mdn = 8.00$) for Group S. However, for Group C, the opposite was true. The W-test showed a statistically significant decrease in *total* difficulty scores, $z = -2.15$, $p = .032$, $r = -.27$ from problem Set 2 ($Mdn = 6.00$, $n = 31$) to problem Set 3 ($Mdn = 6.00$). For Group P, the *total* difficulty score did not significantly change over the two problem sets, $z = -.952$, *ns* with a small effect size ($r = -.13$).

Conclusion. For Group C, the recorded effort levels increased from Set 2 to Set 3. Interestingly, the opposite direction was true for the reported difficulty levels. For Group S and P, the recorded effort levels were roughly equal over the two problem sets. On the other hand, the reported difficulty levels for Group S somewhat increased from Set 2 to set 3.

6.5.4 Comparing the effects of the learning styles

In the following section, the results of the analysis comparing the effects of the learning styles are presented. Table 6.16 shows the mean scores and standard deviations of the dependent measures for the active and reflective learners in the three strategy groups.

Table 6.16: Descriptive statistics for effort and difficulty scores (*total/overall*) for active and reflective in the three strategy groups

	<i>Group S</i>			<i>Group C</i>			<i>Group P</i>		
	<i>n</i>	<i>M</i>	<i>SD</i>	<i>n</i>	<i>M</i>	<i>SD</i>	<i>n</i>	<i>M</i>	<i>SD</i>
Active									
<i>Total</i>									
effort scores (Set 2)	8	6.88	1.64	8	6.00	1.93	8	4.75	1.58
difficulty scores (Set 2)	8	7.00	1.77	7	5.14	1.07	8	6.38	1.77
effort scores (Set 3)	9	6.56	2.30	9	6.56	1.81	6	5.00	1.55
difficulty scores (Set 3)	9	6.89	1.83	9	5.33	1.50	6	7.50	1.64
<i>Overall</i>									
effort scores	8	13.00	3.34	8	12.63	3.54	6	10.00	3.10
difficulty scores	8	13.75	3.62	7	10.43	1.99	6	14.50	2.07
Reflective									
<i>Total</i>									
effort scores (Set 2)	5	5.00	1.58	5	6.40	2.30	6	4.67	1.21
difficulty scores (Set 2)	5	8.20	1.10	6	6.50	2.26	6	6.33	1.51
effort scores (Set 3)	5	6.40	1.14	5	5.80	1.92	5	6.20	1.92
difficulty scores (Set 3)	5	8.20	1.10	5	5.60	1.34	5	7.20	2.28
<i>Overall</i>									
effort scores	5	11.40	2.30	4	12.75	4.35	5	11.00	2.35
difficulty scores	5	16.40	1.14	5	12.20	3.27	5	13.40	3.85

Note: effort scores (1-5); difficulty scores (1-5)

6.5.4.1 The alternative (H4-A1) and (H4-A2) hypotheses

Active learners would perceive their effort (difficulty) as lower (higher) than reflective learners in the *S* strategy (one-tailed).

Active learners would perceive their effort (difficulty) as lower (higher) than reflective learners in the *C* strategy (one-tailed).

Looking at the two different strategy groups separately, the results of the Mann-Whitney U test (see Table 6.17) revealed that the *total* effort/difficulty scores (both sets) and the *overall* effort/difficulty scores of active learners did not differ significantly from reflective learners. In spite of the fact that the U-test results were non-significant, the results showed effect sizes, varying from a small to a medium effect.

Table 6.17a: Results from Mann-Whitney U test investigating *H4-A1*

Group <i>S</i>	
<i>within Group S</i>	
U-test (Act/Ref)	
<i>Total</i>	
effort scores (Set 2)	$U = 8.50, p = .092, r = -.48$ (2-tailed)
effort scores (Set 3)	$U = 20.00, p = .797, r = -.09$ (2-tailed)
difficulty scores (Set 2)	$U = 10.50, p = .169, r = -.40$ (2-tailed)
difficulty scores (Set 3)	$U = 12.50, p = .202, r = -.37$ (2-tailed)
<i>Overall</i>	
effort scores	$U = 14.00, p = .411, r = -.25$ (2-tailed)
difficulty scores	$U = 8.00, p = .087, r = -.49$ (2-tailed)

Note: Significant results are highlighted in bold. Non-significant results are only presented if they provide at least a small effect size of $|r| \geq .05$. The p value represents 2-tailed significance if the result was not in the expected direction.

Table 6.17b: Results from Mann-Whitney U test investigating *H4-A2*

<i>Group C</i>	
<i>within Group C</i>	
U-test (Act/Ref)	
<i>Total</i>	
effort scores (Set 2)	$U = 18.00, p = .418, r = -.08$
effort scores (Set 3)	$U = 16.00, p = .430, r = -.24$ (2-tailed)
difficulty scores (Set 2)	$U = 11.00, p = .139, r = -.42$ (2-tailed)
difficulty scores (Set 3)	$U = 20.00, p = .833, r = -.09$ (2-tailed)
<i>Overall</i>	
difficulty scores	$U = 9.50, p = .216, r = -.38$ (2-tailed)

Note: Significant results are highlighted in bold. Non-significant results are only presented if they provide at least a small effect size of $|r| \geq .05$. The p value represents 2-tailed significance if the result was not in the expected direction.

With regard to data from Group S, the result of the Mann-Whitney U test revealed that the *total* effort scores (Set 2) of active learners did not differ significantly from those of reflective learners, $U = 8.50, z = -1.74, ns$, nevertheless with a medium effect size ($r = -.48$). In particular, the active learners had a higher median score ($Mdn = 6.00, n = 8$) than the reflective learners ($Mdn = 5.00, n = 5$). With respect to data representing the *overall* effort scores, there was no significant difference between the active learners ($Mdn = 13.00, n = 8$) and the reflective learners ($Mdn = 13.00, n = 5$), $U = 14.00, z = -.886, ns, r = -.25$, the active learners had a mean rank of 7.75 while the reflective learners had a mean rank of 5.80.

The result of the Mann-Whitney U test (Group S) revealed that the *total* difficulty scores of the active learners did not differ significantly from the reflective learners, $U = 10.50, z = -1.44, ns$, for Set 2 (Active, $Mdn = 7.00, n = 8$; Reflective $Mdn = 8.00, n = 5$) and $U = 12.50, z = -1.39, ns$, for Set 3 (Active, $Mdn = 7.00, n = 9$; Reflective $Mdn = 8.00, n = 5$). Nevertheless the results show a medium effect size of $r = -.40$ and $r = -.37$, for both Set 2 and Set 3, respectively. With regard to data representing the *overall* difficulty scores, the result of the U-

test was likewise non-significant, $U = 8.00$, $z = -1.78$, *ns*, nevertheless with a medium effect size ($r = -.49$). A closer inspection of the data revealed that the reflective learners ($Mdn = 16.00$, $n = 5$) had a higher median score than the active learners ($Mdn = 14.00$, $n = 8$).

For Group C, the result of the Mann-Whitney U test revealed that the *total* effort scores (Set 3) of the active learners ($Mdn = 7.00$, $n = 9$) did not differ significantly from the reflective learners ($Mdn = 5.00$, $n = 5$), $U = 16.00$, $z = -.882$, *ns*, with a small effect size ($r = -.24$).

The result of the Mann-Whitney U test (i.e. the *total* difficulty scores, Set 2) was likewise non-significant for Group C, $U = 11.00$, $z = -1.51$, *ns*, but with a medium effect size ($r = -.42$). The reflective learners had a slightly higher median score ($Mdn = 6.50$, $n = 6$) than the active learners ($Mdn = 6.00$, $n = 7$). Moreover, looking at the data representing the *overall* difficulty scores, the result of the U test revealed the same tendency, $U = 9.50$, $z = -1.31$, *ns* (Active, $Mdn = 10.00$, $n = 7$; Reflective $Mdn = 14.00$, $n = 5$). In spite of the fact that the result was non-significant, there was a medium effect size ($r = -.38$). The following paragraph provides more discussion of the data representing *overall* scores using boxplots.

Figure 6.4 depicts the distribution of *overall* effort scores, represented in a boxplot. The length of the box in boxplot indicates the interquartile range, which contains 50% of cases (Pallant, 2007 p.76). The line across the boxplot is the median value and that the protruding lines indicate the smallest and largest values of the *overall* effort scores.

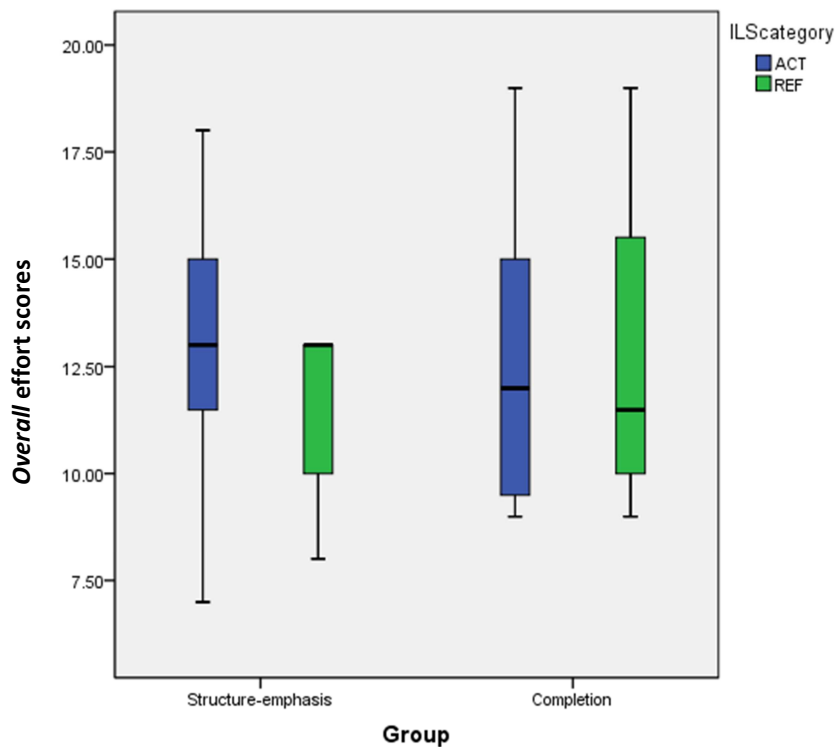


Figure 6.4: Overall effort scores (*Mdn*) for Group S and C, distinguishing between the active and reflective learners

Comparing the active and the reflective learners in Group S, the boxplot in Figure 6.4 shows that both the active ($n = 8$) and the reflective learners ($n = 5$) had just about the same lowest score for *overall* effort. It appears that the median score for the reflective learners was the highest score for the reflective learners. In fact, the median score for the reflective learners and the active learners was at the same level. Finally, the highest score for the active learners was 18.00 while for the reflective learners was 13.00.

The boxplot representing Group C in Figure 6.4 shows that both the active and reflective learners had much the same pattern of scores for *overall* effort. However, the active learners had a median score of 12.00 ($n = 8$) while the reflective learners had a median score of 11.50 ($n = 4$).

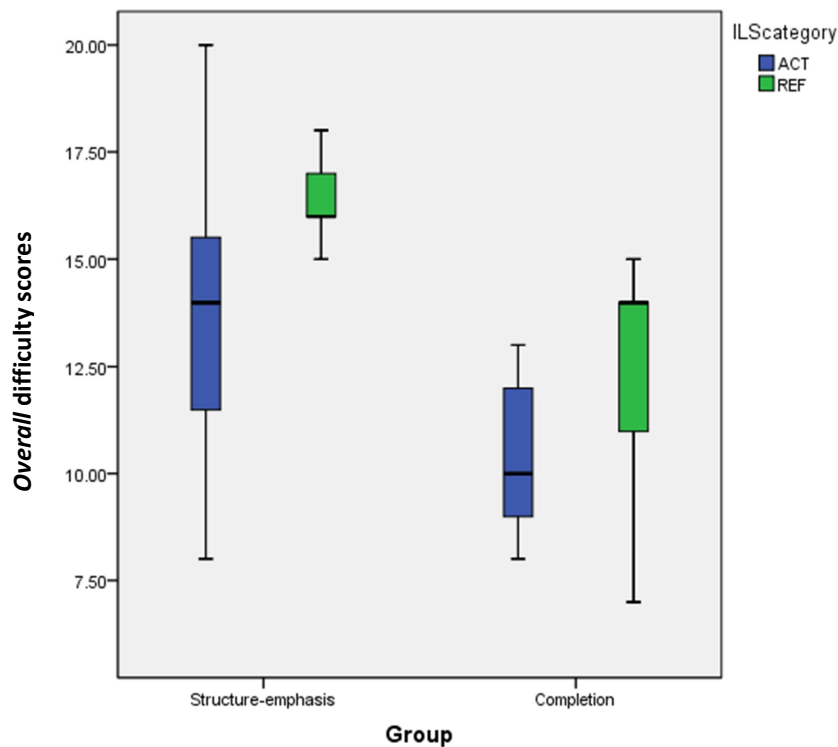


Figure 6.5: Overall difficulty scores (*Mdn*) for Group S and C, distinguishing between the active and reflective learners

The boxplot in Figure 6.5 representing Group S shows that the lowest score for *overall* difficulty of the reflective learners ($n = 5$) was 15.00 while for the active learners ($n = 8$) was 8.00. In fact, the distance between the lowest score of the reflective learners and of the active learners is obvious. However, the highest score for the active learners was 20.00 while for the reflective learners was 18.00. Finally, the median score for the reflective learners was 16.00 while for the active learners was 14.00, which suggests that the reflective learners tended to report higher levels of difficulty than the active learners. Comparing the active and reflective learners in Group C, the boxplot in Figure 6.5 shows that, both the active and the reflective learners had just about the same lowest score for *overall* difficulty. On the other hand, the highest score of the active learners was 13.00 while for the reflective learners was 15.00. The boxplot also shows that the reflective learners had a median score of 14.00 ($n = 5$) while the

active learners had a median score of 10.00 ($n = 7$). In fact, the distance between the median scores of the active and reflective learners is obvious.

Conclusion. A small to a medium effect sizes indicated a trend that the active learners seemed to record higher amounts of effort in studying worked-examples using the Structure-emphasising strategy than the reflective learners. On the other hand, both the active and reflective learners recorded just about the same amounts of effort in studying worked-examples using the Completion strategy. In general, a medium effect size indicated a trend that the reflective learners tended to report higher levels of difficulty than the active learners in both of the strategies. As a final point, the active learners seemed to appear to benefit more with the Structure-emphasising and the Completion strategies than the reflective learners. More specifically, both the strategies seemed to impose low extraneous cognitive load on the active learners as compared to the reflective learners. Thus, the active learners seemed, could potentially invest effort in processes germane for learning. The (alternative) *H4-A1* and *H4-A2* hypotheses were not supported and in fact, contradict the research hypotheses.

6.5.4.2 The null (*H4-01*) hypothesis

Reflective learners would show no difference with respect to their perceived effort and difficulty with the S strategy as compared to the P strategy.

A Mann Whitney U test was performed to observe whether there were significant differences between the reflective learners in Group S and in Group P in terms of perceived effort/difficulty in learning with worked-examples using the Structure-emphasising strategy and the Paired-method strategy, respectively.

Table 6.18: Results from Mann-Whitney U test investigating *H4-01*

<i>Reflective</i>	
<i>between Group S and P</i>	
U-test (Group S/P):	
<i>Total</i>	
effort scores (Set 2)	$U = 13.00, p = .825, r = -.11$
effort scores (Set 3)	$U = 11.00, p = .825, r = -.10$
difficulty scores (Set 2)	$U = 4.00, p = .048, r = -.64$
difficulty scores (Set 3)	$U = 9.00, p = .524, r = -.24$
<i>Overall</i>	
difficulty scores	$U = 6.50, p = .246, r = -.40$

Note: Significant results are highlighted in bold. Non-significant results are only presented if they provide at least a small effect size of $|r| \geq .05$.

The result of the Mann-Whitney U test (see Table 6.18) revealed that the *total* effort scores of the reflective learners in Group S did not differ significantly from those of the reflective learners in Group P, $U = 13.00, z = -.374, ns$ for Set 2 (Group S, $Mdn = 5.00, n = 5$; Group P, $Mdn = 4.50, n = 6$) and $U = 11.00, z = -.319, ns$ for Set 3 (Group S, $Mdn = 6.00, n = 5$; Group P, $Mdn = 6.00, n = 5$). The results also indicated a small effect size of $r = -.11$ and $r = -.10$, for Set 2 and 3 respectively.

The result of the Mann-Whitney U test also revealed that the reflective learners in Group S reported significantly higher levels of difficulty than their reflective counterparts in Group P, Set 2: $U = 4.00, z = -2.11, p = .048, r = -.64$ (Group S, $Mdn = 8.00, n = 5$; Group P, $Mdn = 7.00, n = 6$). Likewise for Set 3, though not significant, $U = 9.00, z = -.747, ns, r = -.24$ (Group S, $Mdn = 8.00, n = 5$; Group P, $Mdn = 7.00, n = 5$). Moreover, with regard to data representing the *overall* difficulty scores (Group S, $Mdn = 16.00, n = 5$; Group P, $Mdn = 14.00, n = 5$), the result of the U-test revealed the same trend, but did not reach significance, $U = 6.50, z = -1.27, ns$, nevertheless with a medium effect size ($r = -.40$).

Conclusion. The reflective learners in Group S and in Group P recorded virtually equal amounts of effort (as indicated by a small effect size) in studying worked-examples using the Structure-emphasising strategy and the Paired-method strategy, respectively. On the other hand, one caveat remained regarding the effect size, that is, a medium and a significantly large effect sizes indicated a trend that the reflective learners in Group S tended to report higher levels of difficulty than their reflective counterparts in Group P. Given such mixed results and insufficient evidence, we failed to reject the (null) *H4-01* hypothesis.

6.5.4.3 The null (*H4-02*) hypothesis

Reflective learners would show no difference with respect to their perceived effort and difficulty with the C strategy as compared to the P strategy.

To test the above hypothesis, a Mann-Whitney U test was performed to see whether there were significant differences in terms of perceived effort/difficulty in learning with worked-examples using the Completion strategy as compared to using the Paired-method strategy.

Table 6.19: Results from Mann-Whitney U test investigating *H4-02*

<i>Reflective</i>	
<i>between Group C and P</i>	
U-test (Group C/P):	
<i>Total</i>	
effort scores (Set 2)	$U = 7.50, p = .197, r = -.42$
effort scores (Set 3)	$U = 10.50, p = .786, r = -.14$
difficulty scores (Set 2)	$U = 17.00, p = .913, r = -.05$
difficulty scores (Set 3)	$U = 7.00, p = .333, r = -.38$
<i>Overall</i>	
effort scores	$U = 8.00, p = .714, r = -.17$
difficulty scores	$U = 10.00, p = .690, r = -.17$

Note: Significant results are highlighted in bold. Non-significant results are only presented if they provide at least a small effect size of $|r| \geq .05$.

No significant difference was found with respect to the *total* effort scores between the reflective learners in Group C and those in Group P, Set 2: $U = 7.50$, $z = -1.40$, *ns*, (Group C, $Mdn = 6.00$, $n = 5$; Group P, $Mdn = 4.50$, $n = 6$); Set 3: $U = 10.50$, $z = -.427$, *ns*, $r = -.14$, (Group C, $Mdn = 5.00$, $n = 5$; Group P, $Mdn = 6.00$, $n = 5$). In spite of the fact that the result of the Mann-Whitney U test was non-significant, the result showed a medium effect size of $r = -.42$ for Set 2. With regard to data representing the *overall* effort scores, the result of the U-test also revealed a non-significant difference between the two groups, $U = 8.00$, $z = -.498$, *ns* (Group C, $Mdn = 11.50$, $n = 4$; Group P, $Mdn = 10.00$, $n = 5$), with an effect size of $r = -.17$.

In spite of the fact that the result of the Mann-Whitney U test (the *total* difficulty scores of Set 3) was non-significant, $U = 7.00$, $z = -1.20$, *ns*, it indicated a medium effect size ($r = -.38$). The result revealed that the reflective learners in Group P ($Mdn = 7.00$, $n = 5$) tended to report higher levels of difficulty than their reflective counterparts in Group C ($Mdn = 5.00$, $n = 5$). Furthermore, with regard to data representing the *overall* difficulty scores, the result of the U-test showed the same trend, but did not reach significance, $U = 10.00$, $z = -.529$, *ns*, $r = -.17$, (Group C, $Mdn = 14.00$, $n = 5$; Group P, $Mdn = 14.00$, $n = 5$).

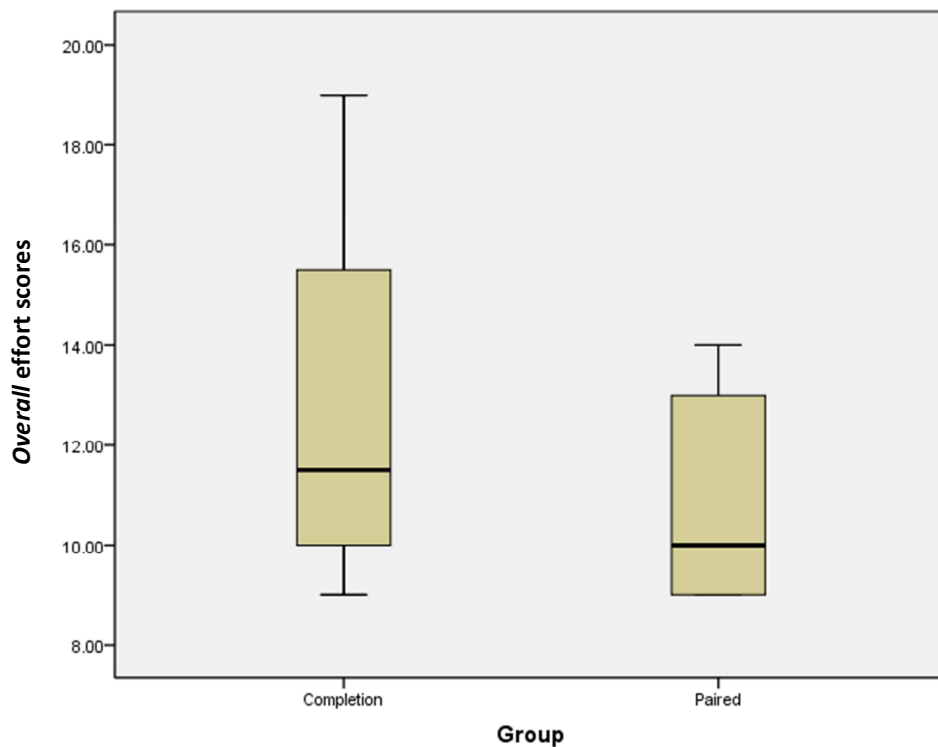


Figure 6.6: Overall effort scores (*Mdn*) of reflective learners in Groups C and P.

The boxplot in Figure 6.6 shows that the reflective learners in Group C and in Group P had the same lowest score for *overall* effort. Nevertheless, the reflective learners in Group C had a median score of 11.50 while the reflective learners in Group P had a median score of 10.00. Overall, the reflective learners in Group C seemed to record higher amounts of effort than the reflective learners in Group P as indicated by the obvious distance between the highest scores of the two groups.

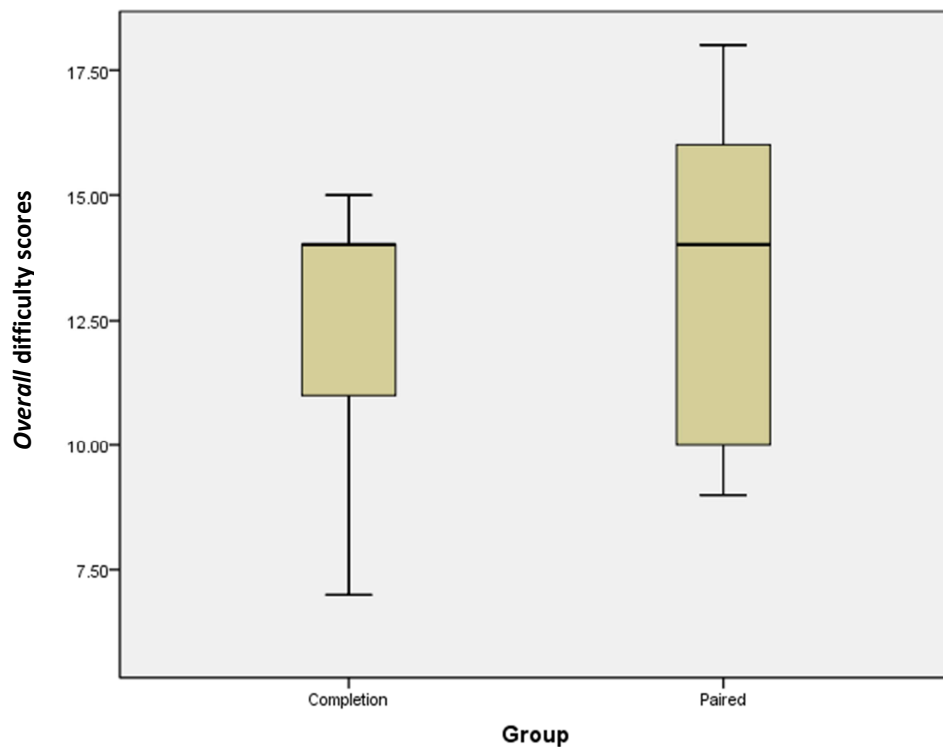


Figure 6.7: Overall difficulty scores (*Mdn*) of reflective learners in Groups C and P.

The boxplot in Figure 6.7 shows that the lowest score for *overall* difficulty of the reflective learners in Group C was 7.00 while for the reflective learners in Group P was 9.00. It appears that the median score for the two groups was at the same level. Overall, the reflective learners in Group C tended to report lower levels of difficulty than the reflective learners in Group P as indicated by obvious distance between the highest scores of the two groups.

Conclusion. One caveat remained regarding the effect size: that is, a small and a medium effect sizes indicated a trend that reflective learners in Group C seemed to record higher amounts of effort and tended to report lower levels of difficulty than their reflective counterparts in Group P. Given these results and insufficient evidence, we failed to reject the (null) *H4-02* hypothesis.

6.5.4.4 The null (H4-03) hypothesis

Active learners would perceive their effort (difficulty) just about or equally high (low) with the P strategy like reflective learners (one-tailed).

A Mann Whitney U test was performed to see whether there were significant differences between the reflective learners and the active learners with respect to their perceived effort/difficulty in learning with worked-examples using the Paired-method strategy.

Table 6.20: Results from Mann-Whitney U test investigating H4-03

Group P	
<i>within Group P</i>	
U-test (Act/Ref):	
<i>Total</i>	
effort scores (Set 3)	$U = 9.50, p = .197, r = -.31$
difficulty scores (Set 2)	$U = 22.50, p = .438, r = -.05$
difficulty scores (Set 3)	$U = 13.00, p = .416, r = -.11$
<i>Overall</i>	
effort scores	$U = 11.00, p = .242, r = -.22$
difficulty scores	$U = 13.00, p = .370, r = -.11$

Note: Non-significant results are only presented if they provide at least a small effect size of $|r| \geq .05$.

Comparing the active and reflective learners in Group P, the results of the Mann-Whitney U test revealed that the *total* effort/difficulty scores and *overall* effort/difficulty scores of the active learners did not differ significantly from those of the reflective learners, see Table 6.20. In spite of the fact that the U-test results were non-significant, the results showed effect sizes, varying from a small to a medium effect.

With regard to the *total* effort scores (of Set 3), $U = 9.50, z = -1.03, ns, r = -.31$, a detailed inspection of the data revealed that the reflective learners ($Mdn = 6.00, n = 5$) had a higher

median score than active learners ($Mdn = 5.00, n = 6$). Moreover, a closer inspection of the data representing the *overall* effort scores revealed the same tendency, $U = 11.0, z = -.737, ns, r = -.22$, that is, the reflective learners ($Mdn = 10.00, n = 5$) had a higher median score than the active learners ($Mdn = 9.50, n = 6$). These results were in the expected direction.

Regarding the *total* difficulty scores (of Set 3), the active learners reported the same median score ($Mdn = 7.00$) as the reflective learners, $U = 13.0, z = -.372, ns, r = -.11$. For the *overall* difficulty scores, $U = 13.0, z = -.368, ns, r = -.11$, active learners ($Mdn = 14.50, n = 6$) had a slightly higher median score than the reflective learners ($Mdn = 14.00, n = 5$).

Conclusion. One caveat remained regarding the effect size, that is, a small and a medium effect sizes indicated a trend that the reflective learners seemed to report higher amounts of effort in studying worked-examples using the Paired-method strategy than the active learners. On the other hand, it appears that the active learners and the reflective learners in Group P reported equal levels of difficulty (as indicated by a small effect sizes) in studying worked-examples using the Paired-method strategy. Given such mixed results and insufficient evidence, we failed to reject the (null) *H4-03* hypothesis.

6.5.4.5 The post-hoc comparison

Are there any differences with respect to overall effort / difficulty scores of active learners in Group P as compared to their active counterparts in Group S and in Group C?

Additionally, *post-hoc* comparisons using the Mann-Whitney U test were performed to examine effort/difficulty in learning with worked-examples using the Paired-method strategy as compared to using the Structure-emphasising strategy and the Completion strategy.

Table 6.21: Results from Mann-Whitney U test of *post-hoc* comparisons

Active	
<i>between Group S (or C) and P</i>	
U-test (Group S/P) * :	
<i>Overall</i>	
effort scores	$U = 12.50, p = .161, r = -.40$
difficulty scores	$U = 21.00, p = .733, r = -.10$
U-test (Group C/P) * :	
<i>Overall</i>	
effort scores	$U = 13.00, p = .178, r = -.38$
difficulty scores	$U = \mathbf{3.00}, p = \mathbf{.008}, r = \mathbf{-.72}$

Note: Significant results are highlighted in bold. Non-significant results are only presented if they provide at least a small effect size of $|r| \geq .05$. * After Bonferroni adjustment (.05/4), significant value is $< .013$

The result of the Mann-Whitney U test (see Table 6.21) revealed that the *overall* effort scores of the active learners in Group S did not differ significantly from their active counterparts in Group P, $U = 12.50, z = -1.50, ns$, however the result indicates a medium effect size ($r = -.40$). A detailed inspection of the data showed that the active learners in Group S ($Mdn = 13.00, n = 8$) seemed to record higher amounts of effort in studying worked-examples using the Structure-emphasising strategy than their active counterparts in Group P ($Mdn = 9.50, n = 6$) using the Paired-method strategy. Similarly, there was no significant difference with

regard to the *overall* difficulty scores between the active learners in Group S ($Mdn = 14.00$, $n = 8$) and in Group P ($Mdn = 14.50$, $n = 6$), $U = 21.0$, $z = -.393$, ns , $r = -.10$.

The result of the Mann-Whitney U test revealed that the *overall* effort scores of the active learners in Group C did not differ significantly from their active counterparts in Group P, $U = 13.0$, $z = -1.43$, ns , nevertheless the result indicates a medium effect size ($r = -.38$). A more detailed inspection of the data showed that the active learners in Group C ($Mdn = 12.00$, $n = 8$) seemed to record higher amounts of effort in studying worked-examples using the Completion strategy than their active counterparts in Group P ($Mdn = 9.50$, $n = 6$) using the Paired-method strategy. On the other hand, there was a significant difference with regard to the *overall* difficulty scores between the active learners in Group C and P, $U = 3.00$, $z = -2.59$, $p = .008$, with a very large effect size ($r = -.72$). A closer inspection of the data shows that the active learners in Group P ($Mdn = 14.50$, $n = 6$) reported higher levels of difficulty than their active counterparts in Group C ($Mdn = 10.00$, $n = 7$).

Conclusion. A medium effect size indicated a trend that the active learners in Group S seemed to record higher amounts of effort in studying worked-examples than the active learners in Group P. However, the active learners in these two groups reported equal levels of difficulty in studying worked-examples. Comparing the active learners in Group C and in Group P, a medium effect size indicated a trend that the active learners in Group C seemed to record higher amounts of effort in studying worked-examples than the active learners in Group P. Finally, active learners in Group P reported higher levels of difficulty than their active counterparts in Group C in studying worked-examples. Table 6.22 and Table 6.23 provide a summary of the conclusions from the analysis comparing the effects of the strategies and the effects of the learning styles, respectively.

Table 6.22: A summary of analysis for comparing the effects of the strategies

	Learning process		
	S/C	S/P	C/P
Recorded <i>effort</i>	Learners in Group S and C spent just about the same amounts of <i>effort</i> in studying worked-examples using Structure-emphasising strategy and Completion strategy, respectively.	Learners in Group S and P spent just about the same amounts of <i>effort</i> in studying worked-examples using Structure-emphasising strategy and Paired-method strategy, respectively.	There was a trend in the data suggesting that learners in Group C seemed to spend more <i>effort</i> in studying worked-examples using Completion strategy than learners in Group P using Paired-method strategy.
Reported <i>difficulty</i>	Learners in Group S reported higher levels of <i>difficulty</i> than learners in Group C in studying worked-examples.	Learners in Group S and P reported equal levels of <i>difficulty</i> in studying worked-examples.	Learners in Group P reported higher levels of <i>difficulty</i> than learners in Group C in studying worked-examples.

Table 6.23: A summary of analysis for comparing the effects of the learning styles

LS	Learning process: between the different strategy groups		
	S/C	S/P	C/P
Act		<p>There was a trend in the data suggesting that active learners in Group S seemed to spend more <i>effort</i> in studying worked-examples using Structure-emphasising strategy than active learners in Group P using Paired-method strategy.</p> <p>Active learners in Group S and P reported equal levels of <i>difficulty</i> in studying worked-examples.</p>	<p>There was a trend in the data suggesting that active learners in Group C seemed to spend more <i>effort</i> in studying worked-examples using Completion strategy than active learners in Group P using Paired-method strategy.</p> <p>Active learners in Group P reported higher levels of <i>difficulty</i> than their active counterparts in Group C.</p>
Ref		<p>Reflective learners in Group S and P spent just about the same amounts of <i>effort</i> in studying worked-examples using Structure-emphasising strategy and Paired-method strategy, respectively.</p> <p>There was a trend in the data suggesting that reflective learners in Group S tended to report higher levels of <i>difficulty</i> than their reflective counterparts in Group P.</p>	<p>There was a trend in the data suggesting that reflective learners in Group C seemed to spend more <i>effort</i> in studying worked-examples using Completion strategy than reflective learners in Group P using Paired-method strategy.</p> <p>There was a trend in the data suggesting that reflective learners in Group P tended to report higher levels of <i>difficulty</i> than their reflective counterparts in Group C.</p>

LS	Learning process: within each of the strategy group		
	S	C	P
Act/Ref	<p>There was a trend in the data suggesting that active learners seemed to spend more <i>effort</i> in studying worked-examples using Structure-emphasising strategy than reflective learners.</p> <p>There was a trend in the data suggesting that reflective learners tended to report higher levels of <i>difficulty</i> than active learners.</p>	<p>Both active and reflective learners spent just about the same amounts of <i>effort</i> in studying worked-examples using Completion strategy.</p> <p>There was a trend in the data suggesting that reflective learners tended to report higher levels of <i>difficulty</i> than active learners.</p>	<p>There was a trend in the data suggesting that reflective learners seemed to spend more <i>effort</i> in studying worked-examples using Paired-method strategy than active learners</p> <p>Both active and reflective learners reported equal levels of <i>difficulty</i> in studying worked-examples.</p>

6.6 Conclusion

This chapter has briefly discussed the methods for statistical data analysis and the main experiment. It has also discussed the analysis results for the Web-OSPAN measures as well as for difficulty and effort measures. In general, the results were inconsistent. In comparing the effects of the three strategies, there were significant differences in reported difficulty and effort during learning, with difficulty but not effort in favour of the Completion strategy. In comparing the effects of the two learning styles, there were no significant differences between active and reflective learners in the three strategy groups on cognitive load measures. Despite this, the reported effect sizes ranging from a medium to a large effect indicated a trend in the data suggesting that the two learning styles seemed to differ in terms of reported effort and/or difficulty. This was also the case when comparing reflective (and active) learners in the three strategy groups.

Chapter 7 Learning *outcome*

7.1 Introduction

This chapter starts with a synopsis of the research hypotheses. The chapter then discusses the analysis of the learning outcomes. The analysis comparing the effect of the strategies and of the learning styles is discussed first, followed by that for the instructional efficiency measures. Next, the chapter discusses the analysis of the differential effects of the learners' prior programming knowledge and lastly provides a conclusion.

7.2 Overview of the research hypotheses for the transfer phase

This section provides an overview of the research hypotheses associated with the transfer phase¹. More specifically, we tested the following hypotheses:

In comparing the effects of the strategies:

1. The Paired-method strategy would lead to better near and far transfer performance than with either the Completion strategy or the Structure-emphasising alone (1-tailed). No prediction was made with regard to the direct comparison between the Structure-emphasising strategy and the Completion strategy (H2).

¹ Note that no specific prediction was made with regard to reported mental effort and time on tests.

In comparing the effects of the learning styles, the following hypotheses² were tested:

2. Active learners would show just about or equal near and far transfer performance like reflective learners in the Paired-method strategy (H5-01).
3. Reflective learners in the Paired-method strategy would show just slightly better than or equal near and far transfer performance as compared to their reflective counterparts in the Structure-emphasising strategy (H5-02) and the Completion strategy (H5-03).
4. Active learners would show worse near and far transfer performance than reflective learners, in both the Structure-emphasising strategy (H5-A1) and the Completion strategy (H5-A2).

7.3 Mental effort scores, time on tests, and post-test scores

The experimental materials for the transfer phase (see Figure 7.1) consisted of 2 sets of two transfer tests, where each test involved a single problem. That is, a set of two near transfer tests (problems) and a set of two far transfer tests (four tests/problems altogether). Numerals represent post-test number (i.e. T1, T2, etc.). The transfer tests (i.e. the post-tests) were administered to participants 2 days after the learning phase was conducted. The post-test problems were identical across the strategy groups. The description of the post-test problems was previously described in Chapter 5. After working with each problem, participants were asked to give an estimation of the mental effort they had invested in solving the problem (see

² The H5 hypothesis in Chapter 3 is broken down into 5 sub-hypotheses: H5-01, H5-02, H5-03, H5-A1, H5-A2

Chapter 5, section 5.3.3.4 on the assessment of cognitive load section). Participants were also requested to write the time at which they had completed the problem on the problem sheet.

Mental effort scores Time on tests Post-test scores of T1	Mental effort scores Time on tests Post-test scores of T2	Mental effort scores Time on tests Post-test scores of T3	Mental effort scores Time on tests Post-test scores of T4
<i>Total</i> mental effort scores <i>Total</i> time on tests <i>Total</i> post-test scores of <u>near transfer data</u>		<i>Total</i> mental effort scores <i>Total</i> time on tests <i>Total</i> post-test scores of <u>far transfer data</u>	
$Z_{\text{ mental effort, } Z_{\text{ post-test scores }}}$ where Z s are standardised scores			

Figure 7.1: Variables for the transfer phase

Prior to the main transfer phase data analysis, the *total* mental effort scores, the *total* time on tests, and the *total* post-test scores were calculated by adding up the reported mental effort, the times, and the post-test scores from each pair of tests, distinguishing the near transfer tests from the far transfer tests. When comparing effort and post-test scores, if one of the mental effort scores associated with one learner was missing, the *total* value was treated simply as missing (nil) and was not replaced by the *mean* value of the remaining mental effort score. The same procedure was applied to the other measured variables. The *totals* were needed in order to investigate any change in the learners' scores between the near and far transfer conditions. The *overall* post-test scores consisted of the addition of the scores from all four of the transfer tests altogether and the variable was used when comparing effects of prior knowledge. The same procedure (as described above) was applied to the variable if there was a missing value. The *overalls* were needed to reflect the accumulated scores of the four transfer tests.

When comparing efficiencies and thus calculating the standardised z scores, the mean mental effort and post-test scores were first calculated by adding up the reported mental effort and post-test scores from all four of the tests and then dividing by the number of scores added. The mean mental effort and post-test scores were then standardised across the three groups, yielding z scores. Note that there were four missing values corresponding to two learners (from Group S and P) on reported mental effort along with post-test score of one of the two far transfer tests (T3 or T4). Thus, each learner's missing values were replaced by the *mean* mental effort and the test score on the remaining test, such as was recommended by Paas and van Merriënboer (1993) i.e. to take into account any missing values.

Exploratory analyses of the data for each group were conducted which consisted of a test of normality using the Kolmogorov-Smirnov test (or K-S test) and a test of homogeneity of variance using Levene's test. Note that Table 7.1 and Table 7.2 provide the output for the tests of normality (and tests of homogeneity of variance) respectively, representing *total* data. The results of the K-S test representing near transfer data for *total* mental effort scores (for Group S and P) and for *total* time on tests (except for Group P) showed that the distributions appeared to be significantly non-normal whereas for *total* post-test scores the distributions appeared to be normal for all groups.

The far transfer data appeared to be significantly non-normal for the *total* mental effort scores (all groups) and for the *total* post-test scores (except for Group C). For the *total* time on tests, the distributions appeared to be normal for all three groups. For both the near and the far transfer data, the results of Levene's test showed that the variances were not significantly different.

Table 7.1: Tests of normality (Transfer phase)

Group		Kolmogorov-Smirnov		
		<i>Statistic</i>	<i>df</i>	<i>Sig.</i>
<i>Total</i> Mental effort (Near)	Structure- emphasising	.196	37	.001
	Completion	.141	36	.068
	Paired-method	.234	37	.000
<i>Total</i> Time on tests (Near)	Structure- emphasising	.155	36	.029
	Completion	.200	35	.001
	Paired-method	.117	37	.200*
<i>Total</i> Post-test scores (Near)	Structure- emphasising	.110	37	.200*
	Completion	.080	36	.200*
	Paired-method	.134	37	.089
<i>Total</i> Mental effort (Far)	Structure- emphasising	.218	36	.000
	Completion	.198	36	.001
	Paired-method	.229	36	.000
<i>Total</i> Time on tests (Far)	Structure- emphasising	.122	36	.196
	Completion	.077	35	.200*
	Paired-method	.098	34	.200*
<i>Total</i> Post-test scores (Far)	Structure- emphasising	.225	36	.000
	Completion	.128	36	.147
	Paired-method	.226	36	.000

Note: * This is a lower bound of the true significance.

Table 7.2: Tests of homogeneity of variance (Transfer phase)

	<i>Levene</i> <i>Statistic</i>	<i>df1</i>	<i>df2</i>	<i>Sig.</i>
<i>Total</i> mental effort (Near)	.376	2	107	.688
<i>Total</i> time on tests (Near)	.493	2	105	.612
<i>Total</i> post-test scores (Near)	.440	2	107	.645
<i>Total</i> mental effort (Far)	.010	2	105	.991
<i>Total</i> time on tests (Far)	1.667	2	102	.194
<i>Total</i> post-test scores (Far)	.271	2	105	.763

The analysis for comparing the effects of the strategies was performed using a Kruskal-Wallis test to explore any differences (for each of the four tests) in terms of their reported mental effort, time on tests, and post-test scores across the three strategy groups. If the result

of the analysis was significant, a follow-up analysis was conducted using non-parametric *post hoc* procedure by means of a Mann-Whitney test.

In addition, an analysis was performed by correlating the reported mental effort (measured on ordinal scales) or time on tests or post-test scores with the ILS values for each of the four transfer tests. These correlations were measured using Spearman's rho. The analysis of correlation was undertaken in order to find out whether there was a relationship, for example, between reported mental effort and ILS values. Following a significant correlation, a further analysis was conducted by comparing the ILS categories over the measured variable using a non-parametric *post hoc* procedure by means of Mann-Whitney test. Comparisons were performed between an active and a reflective preference and also involving data representing a one-directional relationship (whichever was significant), for example, between an active and a balanced preference. The latter was performed in order to find out whether there existed significant differences between the two learning style preferences with respect to mental effort scores, time on tests, or post-test scores. Finally, correlations were also calculated between time on tests (and reported mental effort) and post-test scores to find out if any of the measures correlated with the attainment of scores.

Note that the same set of analyses was repeated (distinguishing near transfer data from far transfer data) for the *total* mental effort scores (this time measured on continuous scales), the *total* time on tests and the *total* post-test scores.

As previously mentioned, the data for the *total* post-test scores (near transfer) were normal and a parametric test was used. That is, an analysis was undertaken using a one-way between-

groups analysis of covariance (ANCOVA) to compare the three different strategy groups on their *total* post-tests scores for the near transfer data, while controlling for the learners' ILS scores. Additionally, a one-way between-groups analysis of variance (ANOVA) was conducted to explore the difference in the *total* post-tests scores between the groups. Moreover, an analysis was undertaken using a Wilcoxon Signed Rank test to find out if any of the strategy groups resulted in, for instance, a decrease in the *total* post-test scores from the near transfer tests to the far transfer tests and if so, the extent to which that strategy promoted the transfer of programming problem solving skills.

Other measures of the relative efficiency of the instructional conditions in terms of the learning *process*, task involvement, and the learning *outcomes* (as discussed in Chapter 5) were also conducted. These measures were performed by comparing the effects of the strategies and by comparing the effects of the learning styles (e.g. between the active learners in the three strategy groups). The learners' efficiency scores were analysed using a one-way ANOVA. Finally, analyses were also conducted to investigate any differential effects of the learners' level of prior programming knowledge in the different worked-example strategies.

The analysis comparing the effects of the learning styles aimed to investigate any differential effects of the three different worked-example strategies on the active learners and on the reflective learners. Practically the same as the analysis for the learning phase, the *comparison* analysis methods were carried out using the Mann-Whitney U test, by measuring the reported mental effort, time on tests, and the post-test scores for each of the four tests as well as by measuring the *total* of each of the variables (distinguishing between the near and far transfer data). The U-test was used to explore the research hypotheses as described in Table 7.3.

Table 7.3: Hypotheses and planned comparisons

Hypotheses	Planned comparison
H5-01: Active learners would show just about or equal near and far transfer performance like reflective learners in the P strategy (one-tailed).	<i>within</i> group: Active compared to Reflective and particularly looking at Group P.
H5-02: Reflective learners in the P strategy would show just slightly better than or equal near and far transfer performance to those of reflective learners in the S strategy (one-tailed).	<i>between</i> the different groups: Group S compared to Group P.
H5-03: Reflective learners in the P strategy would show just slightly better than or equal near and far transfer performance to those of reflective learners in the C strategy (one-tailed).	<i>between</i> the different groups: Group C compared to Group P.
H5-A1: Active learners would show worse near and far transfer performance than reflective learners in the S strategy (one-tailed).	<i>within</i> group: Active compared to Reflective and particularly looking at Group S.
H5-A2: Active learners would show worse near and far transfer performance than reflective learners in the C strategy (one-tailed).	<i>within</i> group: Active compared to Reflective and particularly looking at Group C.

Note: Letters S, C, and P represent strategy format for Group S, Group C and Group P, respectively. S = Structure-emphasising strategy, C = Completion strategy, P = Paired-method strategy that combines both the S and the C strategies.

7.4 Results

An alpha level of .05 was used for all the analyses. A Bonferroni adjustment to the alpha value was applied for the follow-up analysis if any of the results indicated significant differences. As an effect size measure for parametric tests, we used the eta squared η^2 ; that is, values $<.06$ indicate a small effect, values in the range between .06 and .13 indicate a medium effect, and values $>.13$ indicate a large effect (Cohen, 1988). For non-parametric tests, we used the effect size r ; that is values of .1 indicate a small effect, .3 indicate a medium effect, .5 indicate a large effect (Cohen, 1988). Significant results with an effect size are presented and discussed. In addition, non-significant results are presented only if they provide at least an effect size of $|r| \geq .05$ (with the exception of the results of correlation) and discussed if

the results provide at least a small to medium effect size. Note that we conducted multiple tests using Spearman's rho for investigating the correlations between several dependant variables. This may have caused a Type 2 error to occur - getting a significant result by chance. To minimise the possibility of reaching such a wrong conclusion, a further analysis using the Mann-Whitney U test was used to follow up the correlation finding by adjusting the alpha value according to the number of comparisons being made.

Conclusions are based on a significant result with an effect size and also on a non-significant result, but indicated by at least a small to medium effect size. In the latter case, the result merely indicates a trend. Note that the same guidelines laid down in Chapter 6 were used to derive conclusions.

7.4.1 Internal validity

The 5-point of *mental effort* rating scale used in our study revealed good reliability as Cronbach's α coefficient = .77.

7.4.2 Comparing the effects of the strategies

The following paragraphs compare the effects of the strategies. Table 7.4 shows the mean scores and standard deviations on the dependent measures for the three strategy groups.

Table 7.4: Descriptive statistics for mental effort, time on tests, and post-test scores (near and far transfer) for the three strategy groups

	<i>Group S</i>			<i>Group C</i>			<i>Group P</i>		
	<i>n</i>	<i>M</i>	<i>SD</i>	<i>n</i>	<i>M</i>	<i>SD</i>	<i>n</i>	<i>M</i>	<i>SD</i>
<i>Total (Near transfer)</i>									
Mental effort	37	7.19	1.47	36	6.89	1.49	37	6.59	1.46
Time on tests	36	51.58	4.21	35	50.80	4.45	37	51.51	5.31
Post-test scores	37	9.00	5.22	36	10.19	4.55	37	7.97	4.73
<i>Total (Far transfer)</i>									
Mental effort	36	7.89	1.88	36	7.61	1.78	36	7.25	2.02
Time on tests	36	52.53	10.79	35	53.74	7.52	34	54.82	8.05
Post-test scores	36	3.72	4.44	36	4.64	3.68	36	3.58	3.90

Note: Mental effort scores (1-5), post-test scores (0-10)

Table 7.5: Results from Spearman's rho correlation and Mann-Whitney U test (Transfer phase)

<i>Group S</i>	
ILS values and time on tests (Test 3)	$\rho = .379, p = .023$
U-test *	
Ref/Bal	$U = 39.00, p = .631, r = -.09$
Act/Ref	$U = 15.00, p = .710, r = -.13$
ILS values and post-test scores (Test 3)	$\rho = -.379, p = .023$ (2-tailed)
U-test *	
Act/Bal	$U = 61.50, p = .065, r = -.33$
Act/Ref	$U = 10.50, p = .257, r = -.33$ (2-tailed)
Time on test and post-test scores (Test 2)	$\rho = -.399, p = .016$
<i>Total</i> time on tests and <i>total</i> post-test scores (Near transfer)	$\rho = -.534, p = .001$
<i>Group C</i>	
ILS values and reported mental effort (Test 1)	$\rho = .393, p = .018$ (2-tailed)
U-test *	
Act/Bal	$U = 87.00, p = .716, r = -.07$
Act/Ref	$U = 9.50, p = .035, r = -.56$ (2-tailed)
<i>Total</i> time on tests and <i>total</i> post-test scores (Near transfer)	$\rho = -.356, p = .036$
<i>Group P</i>	
Time on test and post-test scores (Test 4)	$\rho = .328, p = .047$
<i>Total</i> time on tests and <i>total</i> post-test scores (Far transfer)	$\rho = .376, p = .029$

Note: Significant results are highlighted in bold. Non-significant results of the U-test are only presented if they provide at least a small effect size of $|r| \geq .05$. * After Bonferroni adjustment (.05/2), significant value is $< .025$. The p value represents 2-tailed significance if the result was not in the expected direction.

7.4.2.1 The analysis of the correlation between ILS values and reported mental effort

The relationship between the ILS values and the reported mental effort (of Test 1) for Group C, measured by Spearman's rho indicates a modest positive correlation, $\rho = .393$, $p = .018$ (see Table 7.5). This correlation indicates that the more active learners tended to report lower levels of mental effort on solving Test 1. Further inspection using the Mann-Whitney U test was used to follow up this finding.

A Bonferroni adjustment to the alpha value was applied and therefore all results are reported at a significance level of .025. It appears that reported levels of mental effort of the learners with an active learning style ($Mdn = 3.00$, $n = 9$) did not significantly differ from those of the learners with a balanced learning style ($Mdn = 3.00$, $n = 21$), $U = 87.00$, $z = -.364$, ns , $r = -.07$. Similarly, the reported levels of mental effort of the active learners did not significantly differ from those of the reflective learners ($Mdn = 4.50$, $n = 6$), $U = 9.50$, $z = -2.16$, ns , nevertheless, there is a large effect size ($r = -.56$).

Conclusion. A large effect size indicated a trend that the reflective learners tended to report higher levels of mental effort in solving Test 1 than the active learners.

7.4.2.2 The analysis of the correlation between ILS values and post-test scores (and time on tests)

The relationship between the ILS values and the post-test scores (of Test 3) for Group S shows a modest negative correlation, $\rho = -.379$, $p = .023$. This correlation indicates that the more active learners tended to achieve higher scores on Test 3. However, further inspection of the data using a Mann-Whitney U test revealed that there was no significant difference in

terms of post-test scores, between those with an active preference ($Mdn = 2.00, n = 9$) and those with a balanced preference ($Mdn = 0.00, n = 23$) $U = 61.50, z = -1.85, ns$ nor between those with an active and those with a reflective preference ($Mdn = 0.00, n = 4$), $U = 10.50, z = -1.20, ns$. Note that, after the Bonferroni adjustment ($.05/2$), the significance value is $< .025$. In spite of the fact that the results of the U-test were non-significant, the effect size was medium ($r = -.33$) for both comparisons.

Looking at the relationship between the ILS values and the time on tests (of Test 3) for Group S, the result of Spearman's rho indicates a modest positive correlation $\rho = .379, p = .023$. This correlation indicates that the more reflective learners tended to spend more time on solving Test 3. Further inspection using the Mann-Whitney U test was used to follow up this finding. A Bonferroni adjustment to the alpha value was applied and therefore all results are reported at the significance level of .025. It appears that the time spent on Test 3 by learners with a reflective learning style ($Mdn = 25.50, n = 4$) did not significantly differ from learners with a balanced learning style ($Mdn = 27.00, n = 23$), $U = 39.00, z = -.481, ns, r = -.09$. Likewise, no significant difference was found between the active learners ($Mdn = 20.00, n = 9$) and the reflective learners, $U = 15.00, z = -.465, ns, r = -.13$.

Conclusion. A medium effect size indicated a trend that the active learners in Group S tended to score higher on Test 3 than the reflective learners. On the other hand, no conclusion can be drawn with respect to the analysis of correlation between the ILS values and the time on the tests.

7.4.2.3 The analysis of the correlation between time on tests and post-test scores

The relationship between time on tests and post-test scores (of Test 2) for Group S, shows a modest negative correlation, $\rho = -.399$, $p = .016$. This correlation indicates that learners in Group S tended to achieve their scores within a shorter amount of time for Test 2.

The relationship between the time on tests and post-test scores (of Test 4) for Group P reveals a modest positive correlation, $\rho = .328$, $p = .047$. This correlation indicates that the longer the time learners spent on solving the test, the higher the scores they tended to achieve. This relation was also supported by the significant result of the correlation involving data representing the *total* time on tests and the *total* post-test scores for the far transfer data (as discussed in the following paragraph).

The analysis of correlation between the *total* time on tests and the *total* post-test scores for the near transfer data (i.e. Test 1 and 2) for Group S and C show a modest negative correlation. This correlation indicates that learners in the two groups tended to achieve their higher scores in the near transfer tests within a shorter amount of time. It appears that the correlation (for Group S) was highly significant, $\rho = -.534$, $p = .001$. But, the correlation (for Group C) was not highly significant, $\rho = -.356$, $p = .036$. In contrast, the relationship between the *total* time on tests and the *total* post-test scores for the far transfer data (i.e. Test 3 and 4) for Group P shows a modest positive correlation, $\rho = .376$, $p = .029$. This correlation indicates that the longer the time the learners spent on solving the far transfer tests, the higher the scores they tended to achieve.

Conclusion. The effect sizes ranging from a medium to a large effect indicated a trend in the data suggesting that learners in Group S and in Group C tended to achieve their scores within shorter amounts of time on the near transfer tests. In contrast, the longer the time learners in Group P spent on solving far transfer test, the higher the scores they tended to achieve.

7.4.2.4 The analysis of results investigating any differences across the strategy groups

The results of a Kruskal-Wallis test revealed that the mental effort, time on tests, and post-test scores for each of the four tests as well as the *total* of each of the variables (distinguishing between near and far transfer tests) did not significantly differ across the three strategy groups. It looks as if solving the near and the far transfer tests required pretty much the same amounts of mental effort in all the strategy groups. Moreover, it seems that the three strategy groups spent virtually the same amounts of time on the near and far transfer tests. Finally, it also appears that the *total* post-test scores on the far transfer tests were just about the same in all three of the strategy groups.

A one-way between-groups analysis of covariance (ANCOVA) was conducted to compare the three different strategy groups on the *total* post-test scores for the near transfer data while controlling for the learners' ILS scores. The independent variable was the strategy group (Group S, C, and P) and the dependent variable consisted of the *total* post-test scores. The learners' ILS scores were used as the covariate. There was no significant difference ($p = .156$) in the *total* post-test scores between the three strategy groups, after controlling for the ILS scores. The effect size indicated a small effect size (partial eta squared = .03). As a final point, there was no significant relationship between the ILS values and the *total* post-test scores, after controlling for the independent variable (i.e. group).

A one-way between-groups analysis of variance (ANOVA) was conducted to explore the difference in the *total* post-test scores for the near transfer data between the three different strategy groups. There was no significant difference, $F(2, 107) = 1.92, p = .152$ in the *total* post-test scores for the three strategy groups. The actual difference in the mean scores between the groups was quite small as described in Table 7.4. The effect size, calculated using eta squared, was small at .03.

Conclusion. The H2 hypothesis that the Paired-method strategy would lead to better near and far transfer performance than either the Completion strategy or the Structure-emphasising was not supported.

7.4.2.5 The analysis of results for repeated measures for mental effort scores and post-test scores

In the following section, an analysis using the Wilcoxon Signed Rank test was conducted to measure any change in scores on the *total* mental effort scale and on the *total* post-test scores across the two sets of transfer tests, see Table 7.6.

Table 7.6: Results from Wilcoxon Signed Ranked test investigating change on the *total* mental effort scores and *total* post-test scores across the two sets of transfer tests (i.e. near and far transfer)

	<i>Group S</i>	<i>Group C</i>	<i>Group P</i>
W-test:			
<i>Total</i> mental effort	$z = -2.27, p = .023, r = -.27$	$z = -2.27, p = .023, r = -.27$	$z = -3.22, p = .001, r = -.38$
<i>Total</i> post-test scores	$z = -5.11, p = .000, r = -.60$	$z = -5.14, p = .000, r = -.61$	$z = -4.90, p = .000, r = -.58$

Note: Significant results are highlighted in bold.

A Wilcoxon Signed Rank test was conducted to compare the *total* mental effort scores invested on the near transfer tests and far transfer tests. For Group S, the *total* mental effort scores invested on the near transfer tests ($Mdn = 7.00, n = 36$) was significantly lower than on the far transfer tests ($Mdn = 8.00$), $z = -2.27, p = .023, r = -.27$. For Group C, the *total* mental effort scores invested on the near transfer tests ($Mdn = 7.00, n = 36$) was likewise significantly lower than on the far transfer tests ($Mdn = 8.00$), $z = -2.27, p = .023, r = -.27$. For Group P, similarly to Groups S and C, the *total* mental effort scores invested on near transfer tests ($Mdn = 6.00, n = 36$) was significantly lower than on the far transfer tests ($Mdn = 7.00$), $z = -3.22, p = .001, r = -.38$.

The same test was also conducted to compare the *total* post-test scores on the near transfer tests and the far transfer tests. For Group S, the *total* post-test scores on the near transfer tests ($Mdn = 8.50, n = 36$) was significantly higher than on the far transfer tests ($Mdn = 3.00$), $z = -5.11, p < .001, r = -.60$. For Group C the *total* post-test scores on the near transfer tests ($Mdn = 10.25, n = 36$) was also significantly higher than on the far transfer tests ($Mdn = 4.00$), $z = -5.14, p < .001, r = -.61$. Likewise for Group P, the *total* post-test scores on the near transfer tests ($Mdn = 7.25, n = 36$) was significantly higher than on the far transfer tests ($Mdn = 2.00$), $z = -4.90, p < .001, r = -.58$.

7.4.3 Comparing the effects of the learning styles

The following section compares the effects of the learning styles. Table 7.7 shows the mean scores and standard deviations of the dependent measures for the active and reflective learners in the three strategy groups.

Table 7.7: Descriptive statistics for mental effort, time on tests, and post-test scores (near and far transfer) for active and reflective in the three strategy groups

	<i>Group S</i>			<i>Group C</i>			<i>Group P</i>		
	<i>n</i>	<i>M</i>	<i>SD</i>	<i>n</i>	<i>M</i>	<i>SD</i>	<i>n</i>	<i>M</i>	<i>SD</i>
Active									
<i>Total</i>									
mental effort scores (N)	9	6.56	1.24	9	6.44	1.94	8	7.50	1.31
time on tests (N)	9	50.56	4.16	9	50.44	5.55	8	51.25	5.18
post-test scores (N)	9	9.22	5.41	9	9.56	5.61	8	6.81	3.88
mental effort scores (F)	9	7.22	1.92	9	7.44	1.42	8	8.63	1.06
time on tests (F)	9	51.56	10.67	9	52.44	6.75	8	54.13	7.04
post-test scores (F)	9	4.56	5.17	9	4.44	4.33	8	2.38	2.33
Reflective									
<i>Total</i>									
mental effort scores (N)	5	7.00	1.58	6	8.00	1.55	6	6.83	0.41
time on tests (N)	4	53.25	5.38	6	51.00	5.06	6	50.33	5.54
post-test scores (N)	5	7.20	8.11	6	11.33	4.24	6	9.00	5.29
mental effort scores (F)	4	7.50	3.00	6	7.67	2.07	6	8.00	0.89
time on tests (F)	4	43.00	17.32	5	59.80	6.50	5	58.20	6.83
post-test scores (F)	4	3.00	6.00	6	5.50	4.18	6	3.67	4.13

Note: mental effort scores (1-5); near / far transfer scores (0-10)

7.4.3.1 The null (H5-01) hypothesis

Active learners would show just about or equal near and far transfer performance like reflective learners in the *P* strategy (one-tailed).

Table 7.8: Results from Mann-Whitney U test investigating *H5-01*

Group <i>P</i>	
<i>within Group P</i>	
U-test (Act/Ref):	
<i>Total</i> (Near transfer)	
mental effort	$U = 16.00, p = .367, r = -.30$
time on tests	$U = 20.00, p = .649, r = -.14$
post-test scores	$U = 16.50, p = .181, r = -.26$
<i>Total</i> (Far transfer)	
mental effort	$U = 16.00, p = .342, r = -.29$
time on tests	$U = 14.00, p = .423, r = -.24$
post-test scores	$U = 21.50, p = .391, r = -.09$

Note: Non-significant results are only presented if they provide at least a small effect size of $|r| \geq .05$.

Table 7.9: Mean rank³ and median for Active and Reflective learners in Group *P*

	<i>Active</i>			<i>Reflective</i>		
	<i>n</i>	<i>Mean rank</i>	<i>Median</i>	<i>n</i>	<i>Mean rank</i>	<i>Median</i>
<i>Total</i> (Near transfer)						
mental effort	8	8.50	7.50	6	6.17	7.00
time on tests	8	8.00	51.50	6	6.83	50.00
post-test scores	8	6.56	6.25	6	8.75	8.00
<i>Total</i> (Far transfer)						
mental effort	8	8.50	8.50	6	6.17	8.00
time on tests	8	6.25	56.50	5	8.20	59.00
post-test scores	8	7.19	2.00	6	7.92	2.00

Comparing the active learners and the reflective learners in Group *P*, the results of the Mann-Whitney U test (see Table 7.8) revealed that the *total* mental effort scores, the *total* time on

³ The U-test converts, for example the post-test scores to ranks across the two learning styles (active and reflective). It then calculates the sum of the ranks for the active and reflective learners separately, and finally evaluates whether the mean rank for the two learning styles differ significantly (Pallant, 2007 p. 220).

tests, and the *total* post-test scores (for both the near and far transfer data) of the active learners did not differ significantly from those of the reflective learners.

Despite the fact that the U-test results for the *total* mental effort were non-significant, the results showed a medium effect size, $r = -.30$ and $r = -.29$ for the near transfer data and the far transfer data, respectively. With regard to the *total* mental effort scores (near transfer), a detailed inspection of the data (see Table 7.9) revealed that the reflective learners ($Mdn = 7.00$, $n = 6$) had a slightly lower median score than the active learners ($Mdn = 7.50$, $n = 8$), $U = 16.00$, $z = -1.11$, *ns*. With regard to the *total* mental effort scores (far transfer), the median score of the reflective learners ($Mdn = 8.00$) was likewise lower than that of the active learners ($Mdn = 8.50$), $U = 16.00$, $z = -1.08$, *ns*.

Regarding the *total* post-test scores for the near transfer tests, there was a tendency in the expected direction and the difference was not significant, $U = 16.50$, $z = -.970$, *ns*, $r = -.26$. A closer inspection of the data revealed that the reflective learners had a median score of 8.00 ($n = 6$) while the active learners had a median score of 6.25 ($n = 8$). For the far transfer tests, the reflective learners had the same median score ($Mdn = 2.00$) as the active learners, $U = 21.50$, $z = -.328$, *ns*, $r = -.09$, though the reflective learners seemed to take longer amounts of time ($Mdn = 59.00$) to solve the far transfer tests than the active ($Mdn = 56.50$), $U = 14.00$, $z = -.882$, *ns*, $r = -.24$.

Conclusion. Despite the fact that the results were non-significant, a medium effect size indicated a trend that the reflective learners tended to report lower levels of mental effort on solving both the near and the far transfer tests than the active learners. On the other hand,

both the reflective and the active learners had fairly equal levels of score on the transfer tests, given that the effect size was small. In view of the instructional efficiency (i.e. a low mental effort combined with a high transfer performance), the reflective learners seemed to benefit more with the Paired-method strategy than the active learners, at least when the near transfer test is considered. A low median score in terms of mental effort of the reflective learners might be attributed to the effort (i.e. germane load) they invested during the learning phase, see Chapter 6, section 6.5.4.4 on *H4-03*. That is, the reported effect size indicated a trend towards a high amount of effort in studying worked-examples by the reflective learners. As a final point, we accept the (null) *H5-01* hypothesis, that the active learners would show just about or equal near and far transfer performance like reflective learners in the P strategy.

7.4.3.2 The null (*H5-02*) hypothesis

Reflective learners in the P strategy would show just slightly better than or equal near and far transfer performance to those of reflective learners in the S strategy (one-tailed).

Table 7.10: Results from Mann-Whitney U test investigating *H5-02*

	Reflective
<i>between Group S and P</i>	
U-test (S/P):	
<i>Total</i> (Near transfer)	
mental effort	$U = 14.00, p = .957, r = -.06$
time on tests	$U = 8.50, p = .500, r = -.24$
post-test scores	$U = 11.00, p = .268, r = -.22$
<i>Total</i> (Far transfer)	
time on tests	$U = 5.00, p = .286, r = -.41$
post-test scores	$U = 7.50, p = .176, r = -.31$

Note: Non-significant results are only presented if they provide at least a small effect size of $|r| \geq .05$.

Table 7.11: Mean rank and median for Reflective learners in Group S and P

	Group S			Group P		
	<i>n</i>	Mean rank	Median	<i>n</i>	Mean rank	Median
<i>Total (Near transfer)</i>						
mental effort	5	6.20	7.00	6	5.83	7.00
time on tests	4	6.38	52.50	6	4.92	50.00
post-test scores	5	5.20	3.00	6	6.67	8.00
<i>Total (Far transfer)</i>						
time on tests	4	3.75	45.00	5	6.00	59.00
post-test scores	4	4.38	0.00	6	6.25	2.00

The result of a Mann-Whitney U test revealed that the *total* mental effort scores of the reflective learners in Group S did not differ significantly from that of the reflective learners in Group P, $U = 14.00$, $z = -.200$, *ns*, $r = -.06$ (see Table 7.10). A detailed inspection of the data revealed that the median score for the two groups was equal (see Table 7.11).

Regarding the *total* post-test scores for the near transfer data, the result of the U-test revealed no significant difference in the expected direction between the reflective learners in Group S and in Group P, $U = 11.00$, $z = -.730$, *ns*, with a small effect size of $r = -.22$. Inspection of the data showed that the reflective learners in Group P had a median score of 8.00 ($n = 6$) while their reflective counterparts in Group S had a median score of 3.00 ($n = 5$). Regarding the *total* post-test scores for the far transfer data, (Group S, $Mdn = 0.00$, $n = 4$; Group P, $Mdn = 2.00$, $n = 6$), the result of the U-test revealed the same trend, and did not reach significance, $U = 7.50$, $z = -.993$, *ns*, nevertheless with a medium effect size of $r = -.31$.

The reflective learners in Group P ($Mdn = 50.00$, $n = 6$) tended to score within shorter amounts of time spent on the near transfer tests as compared to their reflective counterparts in Group S ($Mdn = 52.50$, $n = 4$), $U = 8.50$, $z = -.760$, *ns*, with a small effect size of $r = -.24$. However, the reflective learners in Group P ($Mdn = 59.00$, $n = 5$) seemed to take more time

to solve the far transfer tests than their reflective counterparts in Group S ($Mdn = 45.00$, $n = 4$), $U = 5.00$, $z = -1.23$, ns , $r = -.41$. It should be noted that the reflective learners in Group P spent short amounts of time in solving the near transfer tests, and hence they could use some extra time (more time) for solving the far transfer tests.

Conclusion. Reflective learners in both Group S and P reported equal levels of mental effort on solving the transfer tests. One caveat remained regarding the effect size: a small and a medium effect sizes indicated a trend in the data suggesting that the reflective learners in Group P seemed to score higher on both the near and the far transfer tests than their reflective counterparts in Group S. This might have resulted from difficulty experienced by the reflective learners in Group S during the learning phase. That is, the reported effect size indicated a trend that the reflective learners in Group S tended to report higher levels of difficulty than their reflective counterparts in Group P, see Chapter 6, section 6.5.4.2 on *H4-01*. Finally, we failed to reject the (null) *H5-02* hypothesis, that the reflective learners in the P strategy would show just slightly better than or equal near and far transfer performance to those of reflective learners in the S strategy.

7.4.3.3 The null (H5-03) hypothesis

Reflective learners in the *P* strategy would show just slightly better than or equal near and far transfer performance to those of reflective learners in the *C* strategy (one-tailed).

Table 7.12: Results from a Mann-Whitney U test investigating *H5-03*

<i>Reflective</i>	
<i>between Group C and P</i>	
U-test (C/P):	
<i>Total</i> (Near transfer)	
mental effort	$U = 10.50, p = .318, r = -.39$
post-test scores	$U = 12.00, p = .366, r = -.28$ (2-tailed)
<i>Total</i> (Far transfer)	
time on tests	$U = 11.00, p = .794, r = -.10$
post-test scores	$U = 11.50, p = .331, r = -.30$ (2-tailed)

Note: Non-significant results are only presented if they provide at least a small effect size of $|r| \geq .05$. The p value represents 2-tailed significance if the result was not in the expected direction.

Table 7.13: Mean rank and median for Reflective learners in Group C and P

	<i>Group C</i>			<i>Group P</i>		
	<i>n</i>	<i>Mean rank</i>	<i>Median</i>	<i>n</i>	<i>Mean rank</i>	<i>Median</i>
<i>Total</i> (Near transfer)						
mental effort	6	7.75	8.00	6	5.25	7.00
post-test scores	6	7.50	10.00	6	5.50	8.00
<i>Total</i> (Far transfer)						
time on tests	5	5.80	60.00	5	5.20	59.00
post-test scores	6	7.58	4.00	6	5.42	2.00

No significant difference was found with respect to the *total* mental effort scores (near transfer) between the reflective learners in Group C and in Group P, $U = 10.50, z = -1.35, ns$.

In spite of the fact that the result of the Mann-Whitney U-test was non-significant, the result showed a medium effect size of $r = -.39$ (see Table 7.12). An inspection of the data revealed

that the reflective learners in Group C ($Mdn = 8.00$, $n = 6$) had a higher median score than their reflective counterparts in Group P ($Mdn = 7.00$, $n = 6$), see Table 7.13.

No significant difference was found with respect to the *total* post-test scores (for both the near and far transfer data) between the reflective learners in the two groups. Nevertheless, the results showed medium effect sizes, $r = -.28$ and $r = -.30$ for the near transfer data and the far transfer data, respectively. Regarding the *total* post-test scores (near transfer), $U = 12.00$, $z = -.964$, *ns*, inspection of the data showed that the reflective learners in Group C ($Mdn = 10.00$, $n = 6$) had a higher median score than the reflective learners in Group P ($Mdn = 8.00$, $n = 6$). Regarding the *total* post-test scores (far transfer), $U = 11.50$, $z = -1.05$, *ns*, the median score of the reflective learners in Group C ($Mdn = 4.00$) was likewise higher than that for the reflective learners in Group P ($Mdn = 2.00$).

Finally, the result revealed no significant difference in terms of the time spent on the far transfer tests between the reflective learners in Group C and those in Group P, $U = 11.00$, $z = -.314$, *ns*, with a small effect size of $r = -.10$.

Conclusion. One caveat remained regarding the effect size: a medium effect size indicated a trend in the data suggesting that the reflective learners in Group P tended to score lower on the transfer tests than their reflective counterparts in Group C. This might have resulted from the difficulty experienced by the reflective learners in Group P during the learning phase. That is, the reported effect size indicated a trend that the reflective learners in Group P tended to report higher levels of difficulty than their reflective counterparts in Group C. The reported effect sizes during the learning phase also indicated a trend towards expending a high amount

of effort in studying the worked-examples by the reflective learners in Group C, see Chapter 6, section 6.5.4.3 on *H4-02*. Despite all the above, a medium effect size indicated a trend that the reflective learners in Group C tended to report higher levels of mental effort, particularly on solving the near transfer tests than the reflective learners in Group P. Given these results and insufficient evidence, we failed to reject the (null) *H5-03* hypothesis that the reflective learners in the P strategy would show just slightly better than or equal near and far transfer performance to those of reflective learners in the C strategy.

7.4.3.4 The alternative (*H5-A1*) hypothesis

Active learners would show worse near and far transfer performance than reflective learners in the S strategy (one-tailed).

Table 7.14: Results from Mann-Whitney U test investigating *H5-A1*

Group S	
<i>within Group S</i>	
U-test (Act/Ref):	
<i>Total</i> (Near transfer)	
mental effort	$U = 18.50, p = .675, r = -.15$
time on tests	$U = 12.00, p = .399, r = -.26$
post-test scores	$U = 17.00, p = .500, r = -.20$ (2-tailed)
<i>Total</i> (Far transfer)	
mental effort	$U = 15.00, p = .695, r = -.13$
time on tests	$U = 13.00, p = .503, r = -.21$
post-test scores	$U = 11.00, p = .309, r = -.31$ (2-tailed)

Note: Non-significant results are only presented if they provide at least a small effect size of $|r| \geq .05$. The p value represents 2-tailed significance if the result was not in the expected direction.

Table 7.15: Mean rank and median for Active and Reflective learners in Group S

	<i>Active</i>			<i>Reflective</i>		
	<i>n</i>	<i>Mean rank</i>	<i>Median</i>	<i>n</i>	<i>Mean rank</i>	<i>Median</i>
<i>Total (Near transfer)</i>						
mental effort	9	7.06	6.00	5	8.30	7.00
time on tests	9	6.33	50.00	4	8.50	52.50
post-test scores	9	8.11	8.00	5	6.40	3.00
<i>Total (Far transfer)</i>						
mental effort	9	6.67	8.00	4	7.75	8.00
time on tests	9	7.56	51.00	4	5.75	45.00
post-test scores	9	7.78	3.00	4	5.25	0.00

The results of a Mann-Whitney U-test (see Table 7.14) revealed that the *total* mental effort scores (for both the near and far transfer tests) of the active learners did not differ significantly from those of the reflective learners. With regard to the *total* mental effort scores (near transfer), a detailed inspection of the data (see Table 7.15) revealed that the active learners had a median score of 6.00 ($n = 9$) while the reflective learners had a median score of 7.00 ($n = 5$), $U = 18.50$, $z = -.549$, *ns*. With regard to the *total* mental effort scores (far transfer), the median score of the active learners ($Mdn = 8.00$) was similar to that of the reflective learners, $U = 15.00$, $z = -.473$, *ns*. There were small effect sizes, $r = -.15$ and $r = -.13$ for the near transfer and the far transfer data, respectively.

The results of a Mann-Whitney U-test revealed that the *total* post-test scores for the near transfer tests of the active learners did not differ significantly from those of the reflective learners, $U = 17.00$, $z = -.734$, *ns*, $r = -.20$. Inspection of the data showed that the active learners had a median score of 8.00 ($n = 9$) while the reflective learners had a median score of 3.00 ($n = 5$). Interestingly, the active learners ($Mdn = 50.00$, $n = 9$) tended to score within shorter amounts of time spent as compared to the reflective learners ($Mdn = 52.50$, $n = 4$), $U = 12.00$, $z = -.934$, *ns*, $r = -.26$.

With regard to the *total* post-test scores for the far transfer tests, the result of the U-test revealed the same trend and did not reach significance, $U = 11.00$, $z = -1.11$, *ns*. Though not significant, the result showed a medium effect size of $r = .31$, with the median score of the active learners ($Mdn = 3.00$, $n = 9$) was likewise higher than that for the reflective learners ($Mdn = 0.00$, $n = 4$). However, the active learners ($Mdn = 51.00$, $n = 9$) seemed to take more time to solve the far transfer tests than their reflective counterparts ($Mdn = 45.00$, $n = 4$), $U = 13.00$, $z = -.772$, *ns*, $r = -.21$. It should be noted that the active learners spent shorter amounts of time than the reflective learners in solving the near transfer tests, and hence they could use some extra time (more time) for solving the far transfer tests.

Conclusion. Overall, both the active and the reflective learners reported equal levels of mental effort on solving the transfer tests. On the other hand, a small and a medium effect sizes indicated a trend in the data suggesting that the active learners seemed to score higher on the transfer tests than the reflective learners. Moreover, the active learners tended to achieve their scores within a shorter amount of time spent, particularly on solving the near transfer tests. In view of the instructional efficiency (i.e. a low mental effort combined with a high transfer performance), the active learners seemed to benefit more with the Structure-emphasising strategy than the reflective learners. This would then imply that better performance of the active learners as compared to the reflective learners might be attributed to the effort they had invested during the learning phase. This suggestion is consistent with the trend mentioned in Chapter 6 (see Section 6.5.4.1 on *H4-A1*). That is, the reported effect size indicated a trend that the active learners seemed to spend higher amounts of effort and tended to report lower levels of difficulty in studying the worked-examples using the Structure-emphasising strategy than the reflective learners. Finally, the (alternative) *H5-A1*

hypothesis that the active learners would show worse near and far transfer performance than reflective learners in the S strategy was not supported. In fact the results contradict the hypothesis.

7.4.3.5 The alternative (H5-A2) hypothesis

Active learners would show worse near and far transfer performance than reflective learners in the C strategy (one-tailed).

Table 7.16: Results from Mann-Whitney U test investigating H5-A2

<i>Group C</i>		
<i>within Group C</i>		
U-test (Act/Ref):		
<i>Total (Near transfer)</i>		
mental effort		$U = 14.00, p = .134, r = -.40$
time on tests		$U = 23.00, p = .669, r = -.12$
post-test scores		$U = 23.50, p = .355, r = -.11$
<i>Total (Far transfer)</i>		
mental effort		$U = 23.50, p = .724, r = -.11$
time on tests		$U = 10.00, p = .104, r = -.45$
post-test scores		$U = 22.00, p = .292, r = -.15$

Note: Non-significant results are only presented if they provide at least a small effect size of $|r| \geq .05$.

Table 7.17: Mean rank and median for Active and Reflective learners in Group C

	<i>Active</i>			<i>Reflective</i>		
	<i>n</i>	<i>Mean rank</i>	<i>Median</i>	<i>n</i>	<i>Mean rank</i>	<i>Median</i>
<i>Total (Near transfer)</i>						
mental effort	9	6.56	6.00	6	10.17	8.00
time on tests	9	8.44	51.00	6	7.33	48.50
post-test scores	9	7.61	10.50	6	8.58	10.00
<i>Total (Far transfer)</i>						
mental effort	9	7.61	8.00	6	8.58	8.00
time on tests	9	6.11	53.00	5	10.00	60.00
post-test scores	9	7.44	4.00	6	8.83	4.00

The results of a Mann-Whitney U-test (see Table 7.16) revealed that the *total* mental effort scores, the *total* time on tests, and the *total* post-test scores (for both the near and the far transfer tests) of the active learners did not differ significantly from those of the reflective learners.

With regard to the *total* mental effort scores (near transfer), $U = 14.00$, $z = -1.55$, *ns*, $r = -.40$, a detailed inspection of the data (see Table 7.17) revealed that the reflective learners ($Mdn = 8.00$, $n = 6$) had a higher median score than that of the active learners ($Mdn = 6.00$, $n = 9$). This trend is somewhat supported by the significant result of the correlation between the ILS values and the mental effort scores (of Test 1) in the analysis that compared the effects of the strategies, see Section 7.4.2.1. However, for the far transfer data, the median score of the reflective learners ($Mdn = 8.00$, $n = 6$) was similar to that of the active learners ($Mdn = 8.00$, $n = 9$), $U = 23.50$, $z = -.424$, *ns*, $r = -.11$.

Regarding the *total* post-test scores for the near transfer tests, there was a tendency in the expected direction, but the difference was not significant, $U = 23.50$, $z = -.413$, *ns*, $r = -.11$. Inspection of the data showed that the active learners had a median score of 10.50 ($n = 9$) while the reflective learners had a median score of 10.00 ($n = 6$). With regard to the *total* post-test scores for the far transfer tests, the median score of the active learners ($Mdn = 4.00$, $n = 9$) was similar to that of the reflective learners ($Mdn = 4.00$, $n = 6$), $U = 22.00$, $z = -.591$, *ns*, $r = -.15$. Note that the reflective learners ($Mdn = 60.00$, $n = 5$) seemed to take longer amounts of time to solve the far transfer tests than the active learners ($Mdn = 53.00$, $n = 9$), $U = 10.00$, $z = -1.68$, *ns*, $r = -.45$.

Conclusion. A small and a medium effect sizes indicated a trend that the reflective learners tended to report higher levels of mental effort on solving the transfer tests than the active learners. This might have resulted from the difficulty they experienced during the learning phase. That is, the reported effect size indicated a trend that the reflective learners tended to report higher levels of difficulty than the active learners in studying worked-examples see Chapter 6, Section 6.5.4.1 on *H4-A2*. The reflective learners also seemed to expend greater amounts of time on solving the tests than the active learners. Despite all these, the active learners and the reflective learners had fairly equal levels of score on the transfer tests, given that the effect size was small. As a final point, the (alternative) *H5-A2* hypothesis that the active learners would show worse near and far transfer performance than the reflective learners in the C strategy was not supported. In fact the results contradict the hypothesis.

7.4.4 Instructional efficiency measures

This section describes three different efficiency measures, as introduced in Chapter 5. The use of efficiency measures for comparing the effects of the strategies are discussed first. The use of efficiency measures for comparing the effect of each of the learning styles are discussed next.

Table 7.18: Descriptive statistics for reported difficulty/effort/mental effort, performance, and efficiency means for the three strategy groups

	<i>Group S</i>			<i>Group C</i>			<i>Group P</i>		
	<i>R</i>	<i>P</i>	<i>E/I</i>	<i>R</i>	<i>P</i>	<i>E/I</i>	<i>R</i>	<i>P</i>	<i>E/I</i>
Learning <i>process</i> (LE)	0.37	-0.04	-0.29	-0.43	0.23	0.47	0.04	-0.18	-0.16
Task involvement (INV)	0.18	-0.04	0.10	0.13	0.23	0.25	-0.28	-0.18	-0.33
Learning <i>outcome</i> (IE)	0.20	-0.04	-0.17	0.01	0.23	0.16	-0.19	-0.18	0.01

Note: R represents the Z-scores for either the reported difficulty, effort, or mental effort for learning *process*, task involvement, or learning *outcome*, respectively. P represents the Z-scores for performance on post-tests. E/I represents the efficiency means of either the learning *process* or *outcome* / task involvement, respectively.

7.4.4.1 Learning process efficiency

In terms of the learning process efficiency (LE), there was a statistically significant difference across the three groups: $F(2, 107) = 5.11, p < .01, \eta^2 = .09$. Post-hoc comparisons using the Tukey HSD test revealed that the mean score for Group C ($M = 0.47, SD = 1.02$) was significantly different from that of Group S ($M = -0.29, SD = 1.11$) and from that of Group P ($M = -0.16, SD = 1.10$). But there was no significant difference between Group S and Group P. See Figure 7.2.

7.4.4.2 Task involvement

In terms of task involvement (INV), there was a statistically significant difference across the three groups: $F(2, 107) = 3.10, p < .05, \eta^2 = .05$. Post-hoc comparisons using the Tukey HSD test revealed that the mean score for Group C ($M = 0.25, SD = 1.02$) was significantly different from that of Group P ($M = -0.33, SD = 0.94$). There were no significant difference between Group S ($M = 0.10, SD = 1.12$) and Group P, nor between Group S and Group C. See Figure 7.3.

7.4.4.3 Learning outcome efficiency

There was no significant difference in terms of learning outcome efficiency (IE) scores for the three strategy groups: $F(2, 107) = .87, ns, \eta^2 = .02$. See Figure 7.2 below.

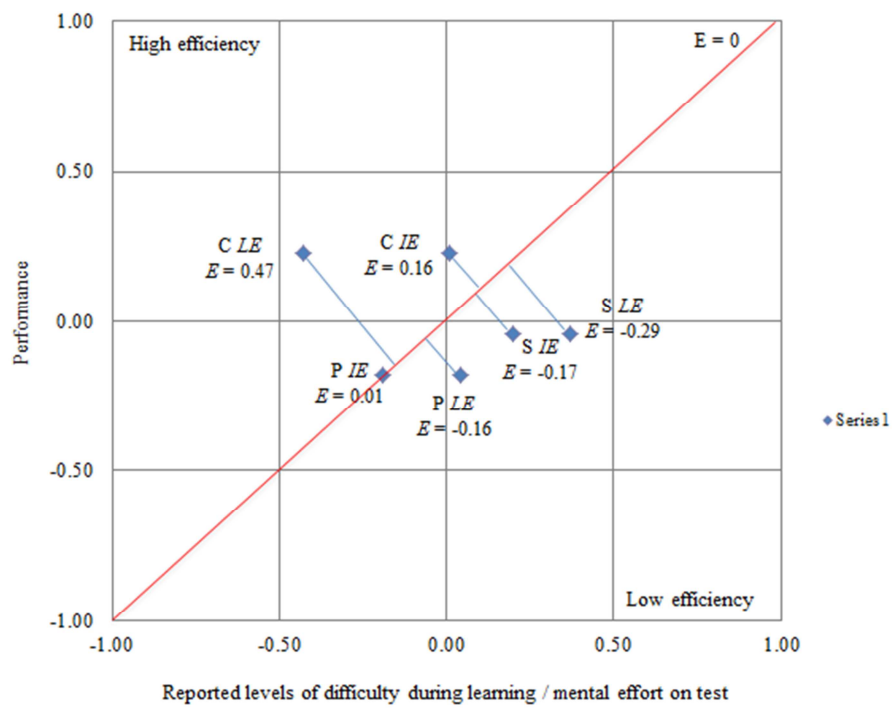


Figure 7.2: Learning *process* and *outcome* efficiencies for the three strategy groups

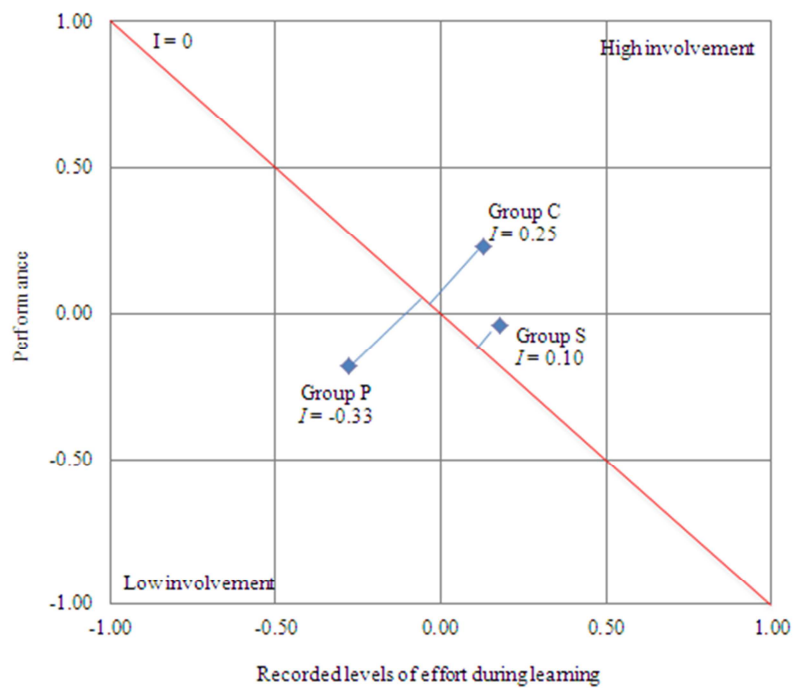


Figure 7.3: Task involvement for the three strategy groups

7.4.4.4 Efficiency measures for active learners in the three strategy groups

Table 7.19: Descriptive statistics for reported difficulty/effort/mental effort, performance, and efficiency means for active learners in the three strategy groups

	<i>Group S</i>			<i>Group C</i>			<i>Group P</i>		
	<i>R</i>	<i>P</i>	<i>E/I</i>	<i>R</i>	<i>P</i>	<i>E/I</i>	<i>R</i>	<i>P</i>	<i>E/I</i>
Learning <i>process</i> (LE)	0.40	0.16	-0.17	-0.53	0.19	0.52	0.16	-0.40	-0.40
Task involvement (INV)	0.46	0.16	0.44	0.13	0.19	0.23	-0.65	-0.40	-0.74
Learning <i>outcome</i> (IE)	-0.30	0.16	0.33	-0.25	0.19	0.32	0.65	-0.40	-0.74

Note: R represents the Z-scores for either the reported difficulty, effort, or mental effort for learning *process*, task involvement, or learning *outcome*, respectively. P represents the Z-scores for performance on post-tests. E/I represents the efficiency means of either the learning *process* or *outcome* / task involvement, respectively.

For the active learners, there were no significant differences for the LE scores, ($F(2, 23) = 1.58$, ns , $\eta^2 = .12$), and the IE scores ($F(2, 23) = 2.75$, ns , $\eta^2 = .19$) across the three groups (see Figure 7.4 for further illustration). But there was a statistically significant difference across the three groups on the INV scores: $F(2, 23) = 3.88$, $p < .05$, $\eta^2 = .25$ (see Figure 7.5). Note that the effect size for the LE scores was medium and that for both the IE scores and the INV scores were large.

Post-hoc comparisons using the Tukey HSD test revealed that the mean INV score for Group P ($M = -0.74$, $SD = 0.66$) was significantly different from that of Group S ($M = 0.44$, $SD = 1.17$). In contrast, Group P did not significantly differ from that of Group C ($M = 0.23$, $SD = 0.84$). Likewise, Group S did not significantly differ from Group C.

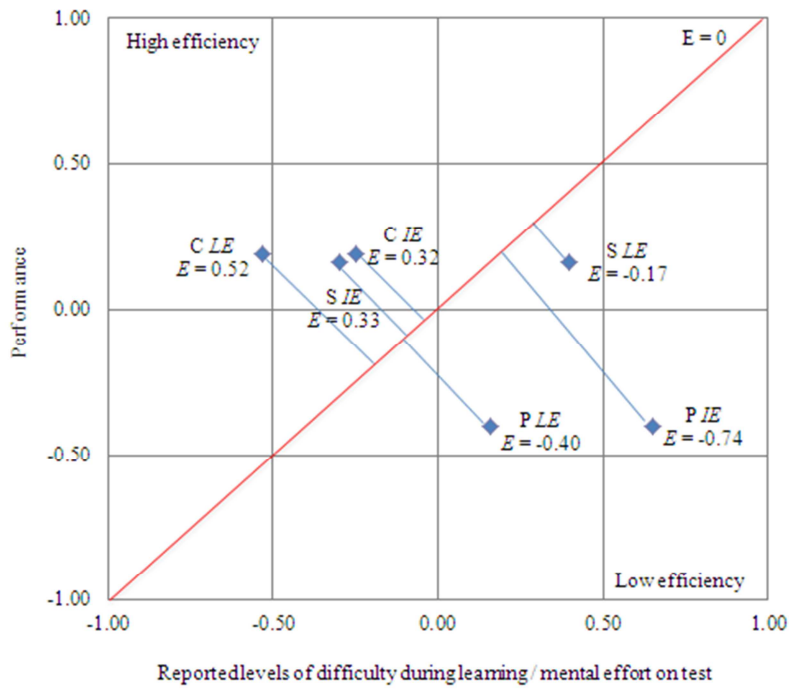


Figure 7.4: Learning *process* and *outcome* efficiencies for the active learners

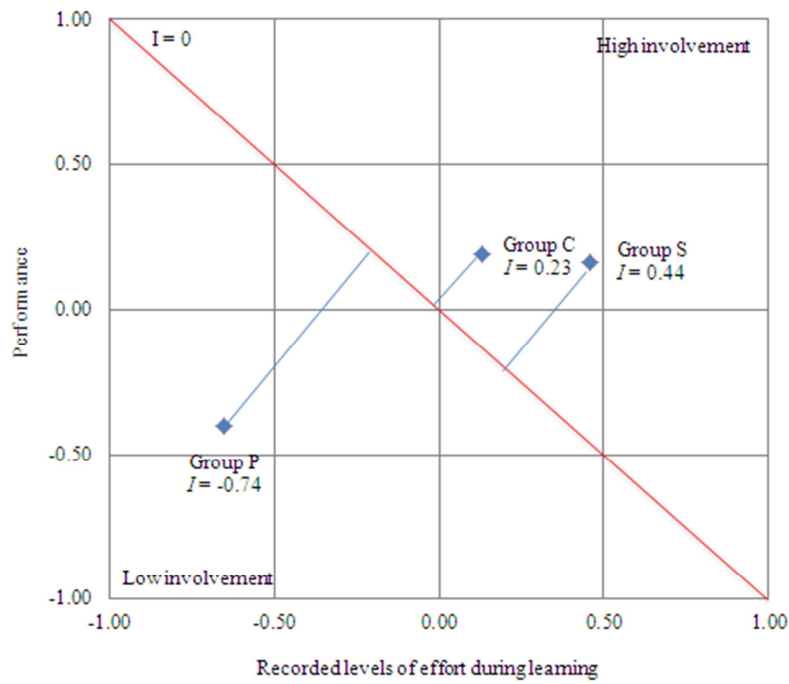


Figure 7.5: Task involvement for the active learners

7.4.4.5 Efficiency measures for reflective learners in the three strategy groups

Table 7.20: Descriptive statistics for reported difficulty/effort/mental effort, performance, and efficiency means for reflective learners in the three strategy groups

	Group S			Group C			Group P		
	R	P	E/I	R	P	E/I	R	P	E/I
Learning <i>process</i> (LE)	0.84	-0.38	-0.86	-0.50	0.37	0.62	-0.19	-0.06	0.09
Task involvement (INV)	0.03	-0.38	-0.25	0.18	0.37	0.40	-0.21	-0.06	-0.20
Learning <i>outcome</i> (IE)	-0.18	-0.38	-0.14	0.22	0.37	0.11	-0.07	-0.06	0.01

Note: R represents the Z-scores for either the reported difficulty, effort, or mental effort for learning *process*, task involvement, or learning *outcome*, respectively. P represents the Z-scores for performance on post-tests. E/I represents the efficiency means of either the learning *process* or *outcome* / task involvement, respectively.

The analysis revealed no significant differences for the LE scores, ($F(2, 16) = 2.41$, ns , $\eta^2 = .26$), INV scores ($F(2, 16) = .56$, ns , $\eta^2 = .07$), or the IE scores ($F(2, 16) = .07$, ns , $\eta^2 = .01$) for the reflective learners across the three groups. Note that the effect sizes for the INV scores and LE scores were medium and large, respectively. See Figure 7.6 and 7.7.

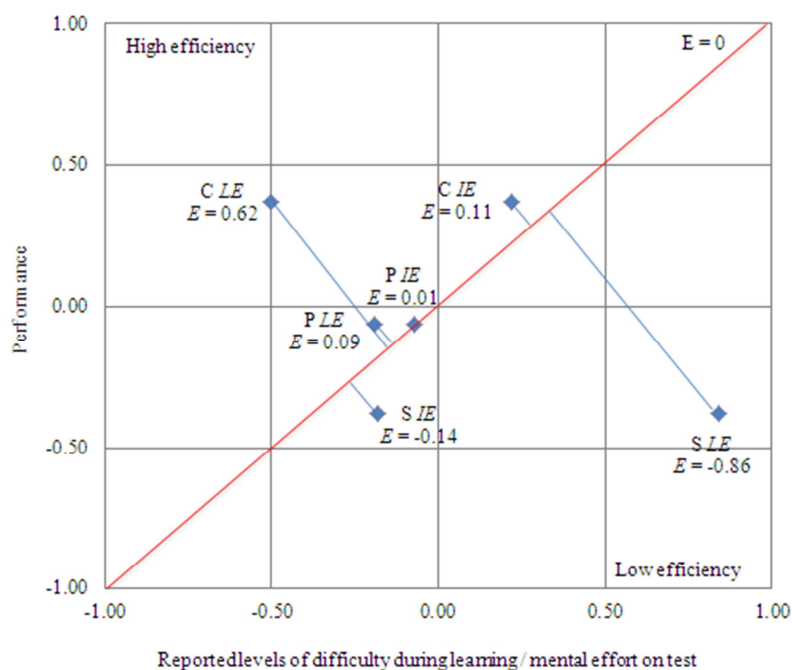


Figure 7.6: Learning *process* and *outcome* efficiencies for the reflective learners

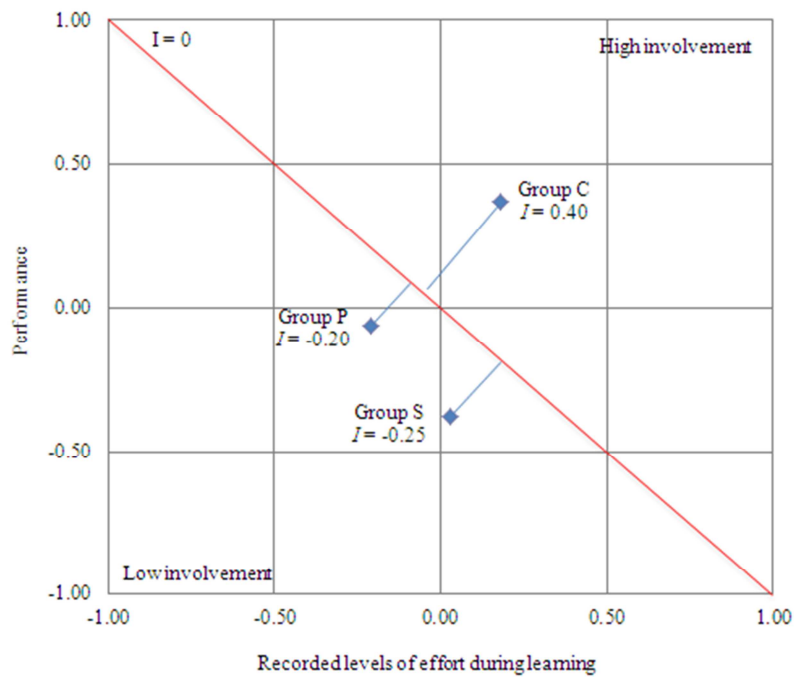


Figure 7.7: Task involvement for the reflective learners

7.4.5 Comparing effects of prior knowledge

The following section compares the effects of prior knowledge for the three strategy groups separately. The level prior programming knowledge was categorised post-hoc into low and high based on the pre-test scores using a median split. Table 7.21 shows the means and standard deviations of the pre-test scores for the low and high prior knowledge learners of the three strategy groups.

Table 7.21: Descriptive statistics for pre-test scores for low and high prior knowledge (post-hoc categorised) in the three strategy groups.

	<i>Group S</i>			<i>Group C</i>			<i>Group P</i>		
	<i>n</i>	<i>M</i>	<i>SD</i>	<i>n</i>	<i>M</i>	<i>SD</i>	<i>n</i>	<i>M</i>	<i>SD</i>
Low prior knowledge	20	5.45	2.11	13	6.23	2.54	27	5.43	2.07
High prior knowledge	15	10.77	1.52	23	10.13	1.36	10	10.35	1.36

Note: Pre-test scores (0-20)

For the high prior knowledge learners, we expected a significant positive correlation (medium to large) to occur between the pre-test scores and the *overall* post-test scores (see Table 7.22). That is, the higher the pre-test scores, the higher the post-test scores they would have been able to achieve, thus providing an indication of the positive effects on learning outcomes. In addition, for the low prior knowledge learners, we expected that they would show positive effects on learning outcomes if they had achieved comparably high levels of performance on post-tests as those of the high prior knowledge learners. Table 7.23 and 7.24 present the analysis of results investigating differences on post-test scores between the low and high prior knowledge learners in each strategy group.

Table 7.22: Results from Spearman's rho investigating correlation between pre-test scores and *overall* post-test scores for the three strategy groups

	<i>Group S</i>	<i>Group C</i>	<i>Group P</i>
Low prior knowledge	$\rho = .68, p = .001$	$\rho = .60, p = .032$	$\rho = .45, p = .022$
High prior knowledge	$\rho = .77, p = .001$	$\rho = -.22, p = .307$	$\rho = .13, p = .724$

Note: Significant results are highlighted in bold.

For Group S, the pre-test scores and the *overall* post-test scores were systematically correlated, that is, significantly large, positive correlations, $\rho = .68, p = .001$ and $\rho = .77, p = .001$ for the low and high prior knowledge learners, respectively. For Group C, there were large, positive ($\rho = .60, p = .032$) and small, negative ($\rho = -.22, ns$) correlations between the two variables for the low and the high prior knowledge learners, respectively. For Group P, there were medium ($\rho = .45, p = .022$) and small ($\rho = .13, ns$) positive correlations between the two variables for the low and the high prior knowledge learners, respectively. In general, these correlations (except for the high prior knowledge learners in Groups C and P) indicate that the pre-test scores were significantly related to the achievement of scores on performance post-test.

The above relationships were further explored using a Mann-Whitney U-test / T-test and conducted separately for each group.

Table 7.23: Results from the Mann-Whitney U test investigating *overall* post-test scores of low and high prior knowledge learners

	<i>Group S</i>	<i>Group P</i>
U-test (Low/High)		
<i>Overall</i> post-test scores	$U = 56.00, p = .003, r = -.52$	$U = 46.00, p = .003, r = -.50$

Note: Significant results are highlighted in bold.

The Mann-Whitney test revealed that there was a significant difference between the *overall* post-test scores between the low ($Mdn = 8.00, n = 19$) and the high ($Mdn = 13.00, n = 15$) prior knowledge learners in Group S, $U = 56.00, z = -3.01, p = .003$, with a large effect size, $r = -.52$. Similarly, there was a significant difference in the *overall* post-test scores between the two categories of prior knowledge learners in Group P (Low, $Mdn = 7.25, n = 26$; High, $Mdn = 15.75, n = 10$), $U = 46.00, z = -2.97, p = .003$, with a large effect size, $r = -.50$.

Table 7.24: Results from T-test investigating *overall* post-test scores on low and high prior knowledge learners

	<i>Group C</i>
T-test (Low/High)	
<i>Overall</i> post-test scores	$t (18.52) = -1.14, p = .269$

An independent-samples T-test was conducted to compare the *overall* post-test scores for the low and the high prior knowledge learners in Group C. There was no significant difference in the scores for the low ($M = 12.73, SD = 9.25$) and for the high ($M = 16.02, SD = 6.34$) prior knowledge learners, $t (18.52) = -1.14, p > .05$ (two-tailed), $\eta^2 = .04$.

A further analysis was conducted to investigate the interaction effects for Group C. The analysis was conducted to see if the influence of pre-test scores on *overall* post-test scores is different for the low and high prior knowledge learners. There was a significant difference in the effect of prior knowledge on the *overall* post-test scores, for the low and high prior knowledge learners, $F(1, 32) = 4.90, p < .05$. Figure 7.8 illustrates the interaction effects for Group C. Note that the two lines are very different in their orientation.

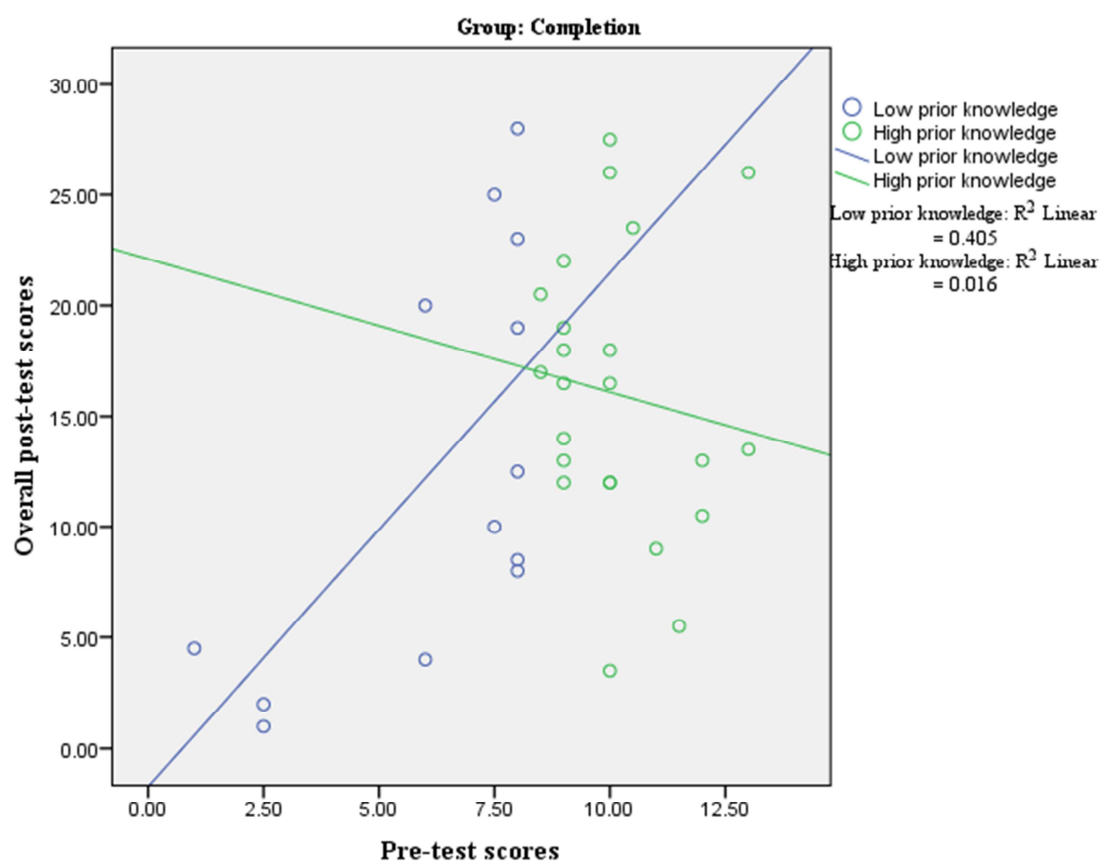


Figure 7.8: Interaction effect in the Completion strategy

Conclusion. In general, the positive correlations between the pre-test scores and the *overall* post-test scores indicated that the learners profited from the strategies, especially the high prior knowledge learners in Groups S (this is further supported by a significant difference in terms of *overall* post-test scores between the low and high prior knowledge learners). By contrast, the significant interaction effect between the pre-test scores and the *overall* post-test scores for Group C suggested a *partial reversal* effect. This suggestion is supported by the finding that there was no significant difference in terms of *overall* post-test scores between the low and the high prior knowledge learners.

7.5 Conclusion

This chapter has briefly discussed the results both for the mental effort measure as well as for the performance on the post-tests. In general, the results were inconsistent. There were no significant differences between the Paired-method strategy and the Completion strategy (nor between the Structure-emphasising strategy) or between the Completion strategy and the Structure-emphasising strategy on reported mental effort and performance post-tests. Likewise, there were no significant differences between active and reflective learners in the three strategy groups (nor between reflective learners in the strategy groups). Despite all this, the reported effect sizes ranging from a medium to a large indicated a trend in the data suggesting that learning styles might have interacted with learners' cognitive load. Table 7.25 and Table 7.26 provide a summary of the analysis for comparing the effects of the strategies and effects of the learning styles, respectively.

Regarding the efficiency measures, the results revealed significant differences between the three strategy groups in terms of the learning *process* and task involvement, with the learning

process in favour of the Completion strategy. Unexpectedly, no significant differences were observed in learning *outcome* efficiencies. On the other hand, there was a trend in the data that suggested a *partial reversal* effect for the Completion strategy and an explanation effect for the Structure-emphasising strategy. Finally, the results revealed no significant differences between reflective learners in the three strategy groups (nor between active learners, with the exception of task involvement).

Table 7.25: A summary of analysis for comparing the effects of the strategies

	Learning outcomes		
	S	C	P
All	Learners in Group S tended to scores within shorter amounts of time spent on solving near transfer tests.	Learners in Group C tended to scores within shorter amounts of time spent on solving near transfer tests.	The longer time learners in Group P spent on solving far transfer tests, the higher the scores they tended to achieve.
	Solving near and far transfer tests required fairly equal amounts of mental effort in all the strategy groups. Moreover, it seems that learners in the three strategy groups tended to spend equal amounts of time on near and far transfer tests. Finally, it appears that post-test scores on near transfer as well as on far transfer were just about the same in all the strategy groups.		
	For all the strategy groups, reported mental effort on near transfer tests was significantly lower than on far transfer tests. It also appeared that post-test scores on near transfer tests were significantly higher than on far transfer tests.		

Table 7.26: A summary of analysis for comparing the effects of the learning styles

LS	Learning outcomes: between the different strategy groups		
	S/C	S/P	C/P
Ref		<p>Reflective learners in Group S and P reported equal levels of mental effort on solving the transfer tests.</p> <p>There was a trend in the data suggesting that reflective learners in Group P seemed to score higher on the transfer tests than reflective learners in Group S.</p> <p><i>Conclusion:</i> reflective learners in Group P seemed to perform better than reflective learners in Group S using the Paired-method strategy and the Structure-emphasising strategy, respectively.</p>	<p>There was a trend in the data suggesting that reflective learners in Group C tended to report higher levels of mental effort, particularly on solving the near transfer tests than reflective learners in Group P.</p> <p>There was a trend in the data suggesting that reflective learners in Group C seemed to score higher on the transfer tests than reflective learners in Group P.</p> <p><i>Conclusion:</i> reflective learners in Group C seemed to perform better than reflective learners in Group P using the Completion strategy and the Paired-method strategy, respectively.</p>

LS	Learning outcomes: within each of the strategy group		
	S	C	P
Act/Ref	<p>Both active and reflective learners reported equal levels of mental effort on solving the transfer tests.</p> <p>There was a trend in the data suggesting that active learners seemed to score higher on the transfer tests than reflective learners.</p> <p><i>Conclusion:</i> active learners seemed to perform better than reflective learners in Group S using the Structure-emphasising strategy.</p>	<p>There was a trend in the data suggesting that reflective learners tended to report higher levels of mental effort on solving the transfer tests than active learners.</p> <p>Both active and reflective learners had fairly equal levels of score on the transfer tests.</p> <p><i>Conclusion:</i> both active and reflective learners seemed to equally perform in Group C using the Completion strategy.</p>	<p>There was a trend in the data suggesting that active learners tended to report higher levels of mental effort on solving the transfer tests than reflective learners.</p> <p>Both active and reflective learners had fairly equal levels of score on the transfer tests.</p> <p><i>Conclusion:</i> reflective learners seemed to perform better than active learners in Group P using the Paired-method strategy.</p>

Chapter 8 Putting the experimental results in context

8.1 Introduction

Recently, Cognitive Load theory (CLT) has been subject to criticism with regard to its conceptual constraints, methodological issues, and practical limitations (e.g. Moreno, 2006, 2010; Schnotz & Kürschner, 2007; de Jong, 2010). A detailed review of these issues is given by de Jong (2010). This chapter discusses the research findings of the thesis in the light of de Jong's criticisms; as such it provides some reflections on the present study.

The chapter starts by briefly discussing the current formulation of cognitive load theory by Sweller (2010). The subsequent sections discuss the present study's research findings, covering issues of self-reporting on cognitive load measures, individual differences, differential effects of the strategies and learning styles, and cognitive load effects, namely a reversal effect and an explanation effect.

8.2 Current formulation of cognitive load theory

In a recent article, Sweller (2010) introduced a formulation that provides a more rational foundation to cognitive load theory, that is, element interactivity is not only linked to intrinsic load but also to extraneous load and *indirectly*, to germane load. In summary, the current formulation suggests that:

“Whereas previously, element interactivity was restricted to providing a common explanatory mechanism for intrinsic cognitive load irrespective of the material being learned, the current work suggests it also can provide an explanatory mechanism for extraneous cognitive load irrespective of the instructional cause of that load. Accordingly, most cognitive load effects, whether based on variations of intrinsic load or extraneous cognitive load, may be explainable using the common concept of element interactivity” (Sweller, 2010 p. 136).

According to Sweller (2010) germane load falls under a different category and differs from intrinsic and extraneous load. Whereas the emphasis of the latter two loads (intrinsic and extraneous) is heavily dependent on the characteristics of the instructional material, germane load, is concerned with the working memory resources that the learner devotes to deal with the element interactivity relevant to the task at hand. Thus, under the current formulation, the total element interactivity underlies intrinsic and extraneous load is the major source of working memory load (i.e. overall cognitive load).

8.3 Self-reporting on cognitive load measures

Earlier cognitive load theory made a distinction between three different types of demand that contribute to total cognitive load, namely intrinsic, extraneous, and germane. It is assumed that these loads are additive (Sweller, van Merriënboer et al., 1998) and it has long been presumed that they are impossible to measure separately. Until recently and under the current formulation, however, overall cognitive load has consisted of the addition of intrinsic and extraneous cognitive load (Sweller, 2010). “Whereas these load types are quite well

distinguishable conceptually, the measurement and empirical separation of them still is one of the major challenges of CLT” (Gerjets, Scheiter & Cierniak, 2009 p. 44).

Recently, several studies have taken a positive step forward in measuring the three different cognitive loads separately. For example, Cierniak, Scheiter et al.’s (2009) study confirmed that the three load types can be disentangled and measured separately through self-reporting, hence providing evidence for the applicability of subjective measures for evaluating different aspects of cognitive load (Gerjets, Scheiter et al., 2009). In a similar vein, Mayer (2009) in his study found it useful to distinguish the three different cognitive loads in terms of underlying cognitive processing, namely extraneous, essential, and generative processing. A study conducted by DeLeeuw and Mayer (2008) partially confirmed this and found that the three cognitive processing types can be detected by different assessment instruments. See Mayer (2010) for a brief overview. Following this line of research, an attempt was made in this study to disentangle two types of cognitive load (i.e. germane and extraneous) to evaluate the effects of three ways of using worked-examples in learning programming and, more importantly, in the hope of shedding some light on the issue of measuring cognitive load.

In our study, cognitive load measures were assessed using two different questions during the learning phase, each on a one-item 5-point rating scale. The question for evaluating germane cognitive load was: “How much new knowledge and skill did you acquire from working on this particular problem?” on a scale ranging from “not at all” to “very much”. The question for evaluating extraneous cognitive load was: “How difficult did you find it to learn things in the recent activity?” on a scale ranging from “very easy” to “very difficult”. Clearly, the questions ask for perceived effort and difficulty, representing germane and extraneous load,

respectively. In particular, the question on effort asks for the extent to which learners had acquired knowledge and skill in the loop concepts from the example problem given, whereas the question on difficulty asks learners to gauge the level of difficulty they faced in the learning process. With regard to the transfer phase, learners had to rate their perceived mental effort in solving the problem on a similar one-item 5-point rating scale, ranging from “very low mental effort” to “very high mental effort”. The following paragraph discusses issues regarding the *reliability* and the *sensitivity* of cognitive load measures in this study.

Concerning the *reliability* of the measure, despite the fact that the perceived effort and difficulty measures in this study showed good internal consistency with Cronbach’s α of 0.79 and 0.75, respectively, there remains some doubt as to whether learners were able to fully understand and distinguish the two questions asked. Even if they did understand, it is not clear how accurately the learners were able to evaluate their perceived effort and difficulty. Moreover, as seen in the learners’ comments, some learners experienced difficulty in distinguishing the difference between the two questions. Variations in learners’ answers were evident in what they regarded as effort and difficulty. More broadly, these questions were answered in relation to several factors (and often mixed them): worked-example problems, program solution (e.g. coding style, algorithm, and variable names), strategy, LECSES interface, and programming background.

This was also the case with the perceived mental effort. A small number of learners in this study regarded low mental effort as being unwillingness to invest effort in solving problems when in actual fact it means needing to use less effort in solving problems (i.e. it is too easy). But note that the results of this study showed internal consistency of Cronbach’s α of 0.77 for

perceived mental effort. Concerning the *sensitivity* of the construct, the results of our study found significant differences across the three strategy groups on perceived effort and difficulty during the learning phase. Unexpectedly, no differences across the groups were detected in the mental effort scale administered during the transfer phase. This was also the case with the performance on post-tests although some treatment effects had been anticipated. In a similar vein, the study conducted by Paas, van Gerven and Wouters (2007) found differences with respect to experienced cognitive load, but not on post-test performance. On the other hand, many studies have indeed found differences with respect to performance on post-tests but they could not detect any differences in cognitive load measures (e.g. Wouters, Paas & van Merriënboer, 2009; de Koning, Tabbers, Rikers & Paas, 2010). An extensive list of such studies is given by de Jong (2010).

Quite simply, these unexpected results used post-test performance as an indirect measure to determine the level of cognitive load expended during learning by attributing better transfer performance as a result of a reduction in extraneous cognitive load and vice versa (de Jong, 2010). More often than not, these results could not be satisfactorily explained on the basis of the theoretical load construct, no plausible reasons were given, and explanations were often highly speculative and without much reference to the empirical evidence from previous cognitive load theory studies (Kirschner, Ayres & Chandler, 2011). Other explanations are offered based on external factors such as motivational level and learning goals (e.g. Seufert, Jänen & Brünken, 2007).

Our study tried to link the result or trend found in the learning phase with that found in the transfer phase to arrive at a plausible explanation for any unexpected results (see the section

below on comparing the effects of the strategies). Furthermore, an attempt is made to build relationships between the empirical evidence and related research findings, and so as to draw firm conclusions.

In conclusion, self-reported ratings of cognitive load measures are highly subjective, thus the question arises: how is the *reliability* of the ratings ensured? Also, in agreement with de Jong (2010), the subjective measure is **not always sensitive** to experimental manipulations and this goes against Paas, van Merriënboer et al.'s (1994) claim. Future research, therefore, should address this issue. Gerjets, Scheiter et al. (2009 p. 51), for instance, recommend researchers "...to develop sets of more sophisticated cognitive-load items and to construct multiple-item rating scales from them that allow distinguishing [and interpreting] different aspects of cognitive load in a consistent way". These items can then be empirically validated across different learning contexts from cognitive processes elicited by them to the learning outcomes, and to the quality of instructional design (Gerjets, Scheiter et al., 2009).

8.4 Individual differences: Learning styles and working memory capacity

Cognitive load research largely employs a between-subjects design to study the effects of different treatment groups on cognitive load (de Jong, 2010). One notable problem of such a design is to maintain equal variances among the treatment groups, which is typically related to individual differences. Every student comes to the learning environment with a different set of characteristics "...such as abilities, interest, or prior knowledge..." (Brunken, Plass et al., 2003 p. 57). Thus, de Jong's (2010 p. 123) recommendation is "...to include these individual characteristics as control variables in experimental set-ups so that differences between groups cannot be attributed to differences in these relevant student characteristics".

According to de Jong (2010), the most common individual characteristic measured and frequently reported to interact with effects on cognitive load is the student's level of knowledge or expertise. That is, "Instructional techniques that are highly effective with inexperienced learners can lose their effectiveness and even have negative consequences when used with more experienced learners" (Kalyuga, Ayres, Chandler & Sweller, 2003 p. 23). See Kalyuga and Renkl (2010) for more recent overviews on expertise reversal effects. We have indeed found a trend in our data (with regard to the Completion strategy) that suggests a *partial reversal* effect, to use Kalyuga and Renkl's (2010) phrase; see the cognitive load effects section for further discussion on this.

Central to cognitive load theory is the idea that "...the design of instructional materials should be aligned with learners' limited cognitive processing resources [i.e. working memory resources] in a way that unnecessary cognitive load is prevented and effective higher-level cognitive processes are supported whenever possible" (Gerjets, Scheiter et al., 2009 p. 43). In this respect, *over(load)*, to use de Jong's (2010) phrase, is determined by the learner's limited working memory capacity. In earlier cognitive load research, however, working memory capacity was seldom measured (de Jong, 2010), apart from the work done by van Gerven, Paas, van Merriënboer and Schmidt (2002; 2004). Recently, several studies have used scores on working memory capacity tests to explain the effects of different treatment conditions (e.g. Lusk, Evans, Jeffrey, Palmer, Wikstrom & Doolittle, 2009; Seufert, Schütze & Brünken, 2009; Berends & van Lieshout, 2009). See de Jong (2010) for an overview of such studies.

Our research has taken a slightly different approach in that the aim was to measure learners' working memory capacity (WMC for short) while taking into account their learning styles, in

the hope of detecting any differences in cognitive *over(load)* related to worked-example strategies on active and reflective learners. It was hypothesised that the learners' learning style is a factor in determining the amount of *serious effort* they are willing to expend on understanding worked-examples, with active learners tending to be more impatient than reflective learners. In this respect, active learners may not be voluntarily motivated to invest effort in learning and accordingly may experience cognitive *over(load)*. This hypothesis was formed following Graf, Lin et al.'s (2008) notion of the *indirect relationship*, which was based on conclusions drawn from various studies. In particular they found that learners with low WMC tend to prefer an active style of learning whereas learners with high WMC tend to prefer a reflective style of learning (Graf, Lin et al., 2008). However, in a recent study conducted by Graf, Liu, Kinshuk, Chen and Yang (2009), they found other significant relationships, in that,

“...the more balanced the learning style is, the higher WMC the learners tend to have. On the other hand, the stronger the preference for either an active or a reflective learning style is, the lower WMC the learners tend to have. Regarding an active learning preference, the results of the experiment are in agreement with the conclusions from the indirect relationship, since both associate low WMC with an active learning preference. However, regarding a reflective preference, conclusions from the indirect relationship argued for high WMC” (Graf, Liu et al., 2009 p. 1286).

Our results¹ revealed no significant association between learning style scores and working memory capacity scores. The absence of significance is disappointing in that it was not possible to replicate Graf, Liu et al.'s (2009) findings. This could be due to several factors associated with the data and the method used in this study, as briefly discussed in Chapter 9. To sum up, scores on working memory capacity (as an independent variable) could not be used for further analysis since the relationship between working memory capacity and learning styles could not be replicated.

Finally, learning styles per se may provide a useful and important parameter in assessing cognitive load. The results of this present study have indeed revealed a trend towards the effects of cognitive load on the active and reflective learners, and this is discussed in the section on comparing the effects of the learning styles. Taken together, individual characteristics such as working memory capacity and learning styles are as important as learner's prior knowledge and together these may influence the effects of cognitive load.

8.5 Comparing the effects of the strategies

We looked at three ways of using worked-examples in learning programming and compared their effects. The Completion strategy required learners to complete examples in which some of the code was missing (e.g. van Merriënboer, 1990). *Structure-emphasising*, to use Quilici and Mayer's (2002) phrase was a strategy that required learners to explain the examples' underlying plan structures. The proposed Paired-method strategy combined the other two strategies. The results of the effects of the strategies are discussed on the basis of three

¹ Note that our experiment was conceived in 2009 and that the research hypotheses were built based on Graf, Lin et al.'s (2008) notion of the *indirect relationship*. Our results will therefore be compared with Graf, Liu et al.'s (2009) recent findings.

efficiency measures (i.e. learning *outcomes*, learning *process*, and task involvement), as already introduced in Chapter 5.

Overall, the learning *outcomes* results showed no significant difference between the three strategy groups although treatment effects for learning *outcomes* had been expected. On the other hand, there were significant differences in efficiency measure expressed in terms of the learning *process* and task involvement. These two measures may provide valuable additional information and may also serve as a basis for drawing an overall interpretation of the findings. The rationale for using effort and difficulty rating in the efficiency measures (i.e. task involvement and learning *process*, respectively) has been discussed in Chapter 5.

In the following subsections, the results are explained according to Sweller's (2010) current formulation. In this study, intrinsic load was kept constant as much as possible across the three strategy groups i.e. without altering what needed to be learned. In other words, variations in cognitive load are defined as a function of different instructional procedures as the element interactivity essential to the to-be-learned materials was constant across the three groups. Under the current formulation:

“If extraneous cognitive load is increased, germane cognitive load is reduced and learning is reduced because the learner is using working memory resources to deal with the extraneous elements imposed by the instructional procedure rather than the essential, intrinsic material” (Sweller, 2010 p. 126).

8.5.1 Completion strategy

Regarding learning *outcome* efficiency, the result does not fit easily with van Merriënboer's (1990b) and Paas's (1992) findings that learning with partly completed examples requires less mental effort and leads to better transfer performance. While van Merriënboer's study compared two learning conditions, program completion versus program generation, Paas's study compared three learning conditions, conventional problem-solving, regular worked-examples and problem completion. Their studies reported superior transfer performance in the completion (and worked-examples) than conventional problem-solving which produced a high extraneous cognitive load. Program/problem completion not only decreased extraneous cognitive load (in comparison to conventional methods) but also increased germane cognitive load (in comparison to regular worked-examples). In contrast, our study compared the effects of using worked-examples in three ways, with all three designed to encourage learners to engage in cognitive processes relevant to schema construction.

Learning with the Completion strategy led to significantly higher efficiency scores in terms of the learning *process* than did learning with the Structure-emphasising and the Paired-method strategies. With regard to relative task involvement, the Completion strategy showed significantly higher efficiency scores than the Paired-method strategy and slightly higher, yet not significantly so than the Structure-emphasising strategy. Despite a lack of significant differences in terms of learning *outcomes* across the three strategy groups, it seems plausible to conclude that the Completion strategy was more efficient than either the Paired-method strategy or the Structure-emphasising strategy and this evidence comes from the efficiency measures (learning *process* and task involvement) which were significant. That is, higher efficiency in terms of the learning *process* and task involvement were **positively related** to

higher learning *outcome* efficiency in the Completion strategy, and this provides an indication of better learning performance.

Thus the Completion strategy appears to have directed attention to processes relevant to schema acquisition, i.e. it promoted “mindful abstraction” (van Merriënboer & Paas, 1990 p. 279) and supported self-explanation (Stark, 1999, as quoted in Renkl, Atkinson et al. 2000). As a final point, we have indeed found a trend towards a *partial reversal* effect (Kalyuga & Renkl, 2010). This effect will be discussed later in the cognitive load effects section.

8.5.2 Paired-method strategy

Following Soloway (1986), the Paired-method strategy combined the Structure-emphasising strategy that requires explanation/reflection, with the Completion strategy that dictates attention to mechanism (i.e. how an incomplete solution should be completed/modified). Using cognitive load theory, it is argued that the Structure-emphasising strategy increases germane cognitive load because learners are guided by plan-focused self-explanation prompts. A similar benefit should be achieved with the Completion strategy. In line with the notion of “worked-out examples function as *analogies*” (van Merriënboer & Krammer, 1987 p. 267), the study of worked-examples using the Structure-emphasising strategy should enable learners to solve the completion task without much difficulty by mapping this task with existing schemata, hence drastically decreasing extraneous cognitive load.

Moreover, the strategy directs cognitive resources (i.e. germane cognitive load) to deal with the interacting elements, and as a consequence strengthens the construction of cognitive schemata. These activities may add an extra processing load to working memory, but it is still

germane to the task at hand and hence should further enhance transfer. In line with Anderson's (1987) theory of knowledge compilation and Soloway's (1986) notion, the Paired-method strategy should at least be as efficient (in terms of learning *outcomes*), if not more so, than the Completion strategy.

Significantly lower efficiency scores for both the learning *process* and task involvement (in comparison to the Completion strategy) may have contributed to little effect on transfer in the Paired-method strategy. Learners were considerably affected by the *pairing sequence*; perhaps they were not comfortably ready to switch from the Structure-emphasising strategy to the Completion strategy. Their learning was somewhat hampered rather than facilitated and this imposed a high demand on working memory (as indicated by significantly higher levels of reported difficulty in studying worked-examples than the Completion strategy). Another possible explanation may be attributable to motivational factors. For example, high levels of reported difficulty may have suggested that "...learners have difficulties learning because they are overwhelmed by the task complexity" (van Merriënboer, Kirschner et al., 2003 p. 5) and as a result they may not have been sufficiently motivated to invest effort in learning (Paas, Tuovinen et al., 2005). This evidence came from the task involvement score for the Paired-method strategy which was significantly lower than the Completion strategy, as mentioned above.

Overall, the results support Paas and van Merriënboer's (1994a) argument that instructional procedures have little effect on transfer unless learners are motivated and willing to invest effort in learning. As a final point, it can be argued that the Paired-method strategy resembles the study conducted by Reisslein, Atkinson, Seeling and Reisslein (2006). In particular, the

Paired-method strategy will be compared to the instructional procedure emphasising example-problem pairs in Reisslein, Atkinson et al.'s study and this will be discussed in the cognitive load effects section.

8.5.3 Structure-emphasising strategy

Very low learning *outcome* efficiency for the Structure-emphasising strategy may be explained by the fact that the strategy was relatively new for these learners. The strategy required learners to construct textual explanations of various instances of plan structures in each example solution, guided by plan-focused prompts and hints. However, these learners had not been trained explicitly to explain and reflect upon the learning materials using plans. Learning with this strategy may have appeared to be demanding for these learners thus resulting in an unnecessary load imposed on their working memory. This was evident from significantly higher levels of reported difficulty in studying worked-examples in the Structure-emphasising strategy than in the Completion strategy. Moreover, evidence came from significantly lower scores in terms of the learning *process* in the Structure-emphasising strategy as compared with the Completion strategy. In this instance, the extraneous cognitive load factor may have impeded schema acquisition in the Structure-emphasising strategy. Moreover and in line with the knowledge compilation theory (Anderson, 1987), learners in this strategy could only acquire declarative knowledge and to a lesser extent, procedural understanding (given that they have been asked to construct an explanation largely on procedural aspects of the example solution) however, they could not potentially acquire problem solving rules since they were not given an opportunity to solve completion problems (i.e. applying knowledge into practice) as did their counterparts in the Completion and Paired-method strategies.

Despite the fact that the Structure-emphasising strategy appeared to exhibit higher task involvement scores than the Paired-method strategy, yet slightly lower than the Completion strategy, learners in this strategy may have invested **ineffective effort** in generating their explanations so that their learning was hampered. Consequently, less cognitive capacity was free for actual learning to take place, i.e. schema abstraction. Another possible explanation is that the poor quality of the learners' explanations indicated that they had not fully understood the examples. This is likely to have affected the organisation and accessibility of their schemata (Bassok & Holyoak, 1989) and thus resulted in poor transfer performance.

It can be argued that the Structure-emphasising strategy used in our study resembles studies on the effect of explanation prompts (Berthold, Röder, Knörzer, Kessler & Renkl, 2011; Berthold, Eysink & Renkl, 2009; Berthold & Renkl, 2009) and process-oriented worked-examples (van Gog, Paas & van Merriënboer, 2006; 2008). The result of our study partially corresponds with that of Berthold and Renkl's (2009) study and will be discussed in relation to the explanation effect (in the cognitive load effects section).

To conclude, we note one important point of concern, as van Gog and Paas (2008, p. 20) wrote:

“Mental effort in combination with performance measures [i.e. efficiency measure in terms of learning *outcomes*] will provide us with a better, more subtle indicator of the quality of learning outcomes, that is, in terms of the efficiency of cognitive schemata acquired, elaborated, or automated as a result of instruction, and hence with a better indicator of the quality of different instructional conditions.”

That is to say, the efficiency measure provides a more sensitive indicator of the quality of learning *outcomes* than just performance tests scores on their own as it takes account of the mental effort in using them. According to van Gog and Paas (2008), the more (less) efficient cognitive schemata acquired is determined by a high (low) performance combined with low (high) mental effort on tests. However, this seems unlikely in certain experimental situations. For example, in our study, it was rather difficult to interpret the efficiency (in terms of learning *outcomes*) of the Structure-emphasising strategy in relation to the efficiency of the Paired-method strategy (refer to Chapter 7, section 7.4.4, Table 7.18). More specifically, the former strategy showed a high level of performance combined with a high level of invested mental effort whereas the latter strategy showed a low level of performance combined with a low level of invested mental effort, which has led to **unclear experimental situations**, to use de Jong's (2010) phrase. This issue has also been acknowledged by Moreno and Valdez (2005) in their study, comparing Group I (which used interactive strategies) and Group NI (which did not use interactive strategies) and states that a relatively higher performance combined with a higher cognitive load (Group I condition) and that a relatively lower performance combined with a lower cognitive load (Group NI condition), it "may lead to equivalent intermediate efficiency conditions" (Moreno & Valdez, 2005 p. 37).

In trying to interpret the result, we note that "...the efficiency measure does not reflect the importance of performance; that is, it does not reflect absolute performance levels" (Paas & van Merriënboer, 1993 p. 742). The efficiency scores for learning *outcomes* should be accompanied by performance test scores, and together they provide a more subtle indicator of the quality of learning *outcomes* than just relying on the test scores. Summing up, it may be concluded that in terms of the quality of the learning *outcomes* (i.e. the quality of cognitive

schemata acquired), the Paired-method strategy was no more efficient than the Structure-emphasising strategy.

8.6 Comparing the effects of the learning styles

Looking at the active and reflective learners in each group separately, the results of instructional efficiencies (in terms of the learning *process*, task involvement, and learning *outcomes*) were somewhat at variance from what was initially predicted.

First, it was expected that active learners would engage well with the Paired-method strategy and, conversely, would do worse in either of the Completion or the Structure-emphasising strategies. In terms of the learning *process*, there was a trend in the data across the groups suggesting that the Completion strategy tended to achieve higher efficiency scores (statistically non-significant, but with a medium effect size) than the Structure-emphasising and the Paired-method strategies, with the latter showing significantly higher levels of reported difficulty in studying worked-examples than the Completion strategy. On the other hand, the task involvement scores for the Paired-method strategy were significantly lower at $p < .05$ level (as indicated by a medium effect size, calculated using eta squared) than the Structure-emphasising strategy. There was a trend in the data across the groups suggesting that whilst active learners in the Completion and the Structure-emphasising strategies both attained fairly equal levels of performance on the learning *outcomes*, active learners in the Paired-method strategy tended to perform much lower than the former strategies (statistically non-significant; however, indicated by a fairly large effect size).

Second, it was hoped that reflective learners using the Paired-method strategy would perform just slightly better than, if not equal to those of their reflective counterparts using either of the single methods on their own. In general, the results of the efficiency measures for the reflective learners across the three strategy groups were inconsistent. With regard to the learning *process* efficiency, there was a trend in the data across the groups suggesting that the Completion strategy tended to achieve higher efficiency scores (statistically non-significant, but with a large effect size) than the Paired-method strategy and the Structure-emphasising strategy, with the latter tended to score very low efficiency scores. Note also, that reflective learners in the Structure-emphasising strategy reported significantly higher levels of difficulty in studying worked-examples than their reflective counterparts in the Paired-method strategy. No significant difference was observed with regard to task involvement scores; nevertheless, a trend in the data across the groups suggesting that the Completion strategy tended to achieve higher scores than both the Paired-method and the Structure-emphasising strategies, as indicated by a medium effect size. In terms of the learning *outcomes*, the results matched expectations. There was no significant difference across the groups, as indicated by a small effect size. Then again, reflective learners in the Structure-emphasising strategy scored far lower than their counterparts in the other two strategies.

Overall, our results showed that both the active and the reflective learners seemed to benefit equally from the Completion strategy; this is at variance with the findings of van Merriënboer's (1990a) study. In a future study, it would be interesting to investigate the completion effect in a different domain, taking into account learning styles. Moreover, there is some doubt whether the reflective learners in the Structure-emphasising strategy exhibited a preference for an active style. Hence, more studies are clearly called for to validate these

findings. Finally, it should be noted that the sample size in our study was not large enough to attain statistical significance; nevertheless, the effect size associated with the results revealed effects ranging from a medium to large effect size. This suggests that learning style may have interacted with effects on learners' cognitive load. Future research should be conducted with larger sample sizes to substantiate this claim.

8.7 Cognitive load effects

Despite observing no difference in the learning *outcomes*, we found a trend in our data (in the Completion strategy) that suggested a *partial reversal* effect (Kalyuga & Renkl, 2010). An expertise reversal effect occurs when an instructional procedure essential for novice learners becomes less effective for more experienced learners (Kalyuga & Renkl, 2010; Sweller, 2010; Kalyuga, Ayres et al., 2003). Our results also partially replicated findings obtained in the Berthold, Eysink et al.'s (2009) study on the explanation effect. In particular, this section further discusses the effects of the strategies on learning *outcomes* associated with two cognitive load effects, namely the expertise and/or *partial reversal* effect and the explanation effect of high prior knowledge and low prior knowledge learners. We reviewed recent empirical findings in trying to interpret our results and draw firm conclusions.

It should be noted that our experimental procedure differed from cognitive load theory inspired experiments that aim to investigate expertise reversal effect on instructional procedures of high and low prior knowledge learners. In contrast, our procedure was initially designed to investigate the effects of learning via three different worked-examples strategies on schema acquisition and transfer of active and reflective learners. As described in Chapter 7, for high prior knowledge learners, we expected a significant positive correlation to occur

between pre-test scores and *overall* post-test scores. That is, the higher the pre-test scores, the higher the post-test scores that should have been achieved, thus providing an indication of positive effects on learning *outcomes*. Besides, it was assumed that low prior knowledge learners would show positive effects on learning *outcomes* if they had achieved comparably high levels of performance as those of the high prior knowledge learners.

With regard to the Completion strategy, the results revealed no significant difference in terms of *overall* post-test scores between high prior knowledge and low prior knowledge learners. Indeed there was a significant interaction effect at the $p < .05$ level between prior knowledge and *overall* post-test scores (as previously discussed in Chapter 7, see Figure 7.8). Moreover, in contrast with what was initially expected, the results of the correlation analysis for the high prior knowledge learners indicated a small, negative correlation ($r = -.22$) between pre-test scores and *overall* post-test scores. However, this correlation was not significant. These trends suggest that the strategy had started to hamper learning for the high prior knowledge learners, thus providing some indication of a *partial reversal* effect, (see Kalyuga & Renkl, 2010). In other words, as learning progressed, the strategy had a reverse effect on the high prior knowledge learners. These learners may have become overwhelmed and experienced extraneous cognitive load because they may not have been able to disregard redundant information. Although, these learners should have been able to “integrate and cross-reference the redundant information with their available knowledge schemas” (Kalyuga, Ayres et al., 2003 p. 9), the act of integrating and cross-referencing itself may have added an extra processing load to their limited working memory capacity. Thus, under the current cognitive load theory formulation, an increase in extraneous cognitive load will reduce germane cognitive load as cognitive resources are being devoted to processing extraneous elements more so than essential, intrinsic materials.

As with the Paired-method strategy, high prior knowledge learners appeared to benefit from the potential effect of the strategy more than low prior knowledge learners. In particular, it was found that the *overall* post-test scores of learners with high prior knowledge were significantly higher than those of learners with low prior knowledge (as indicated by a large effect size). Moreover, looking at these two groups separately, the pre-test scores for the low prior knowledge learners were somewhat systematically correlated with the *overall* post-test scores, as indicated by a significantly medium, positive correlation at $p < .05$ level; in contrast, the result of the correlation analysis for high prior knowledge learners indicated a small, positive correlation ($r = .13$), though not significant. Note that, this direction was the opposite of that for their high prior knowledge counterparts in the Completion strategy. Based on this evidence and in the light of the current cognitive load theory formulation, it can be argued that the high prior knowledge learners had yet to reach the level of knowledge or expertise necessary to show a full reversal effect. On the other hand, the positive effects of the learning *outcomes* did not seem to show up for the low prior knowledge learners. Our results contrast with the study conducted by Reisslein, Atkinson et al. (2006). In particular, they found that low prior knowledge learners performed comparably in terms of learning performance to high prior knowledge learners when presented with the same procedure, i.e., the example-problem pairs. Besides, Reisslein, Atkinson et al. (2006) observed that high prior knowledge learners benefited more from the problem-example pairs than their low prior knowledge counterparts.

Several studies failed to find any positive effects of instructional explanation and/or self-explanation prompts on learning and transfer (van Gog, Paas et al., 2008). For example, Gerjets, Scheiter and Catrambone (2006) looked at the effects of instructional explanations

(Experiment 1) and self-explanation prompts in learning (Experiment 2) in either molar or modular formats in the domain of probability. The molar format treats complex solutions as the basic unit that cannot be broken down further; in contrast, the modular format focuses on breaking down complex solutions into smaller meaningful elements, thus allowing only a limited number of elements to be processed simultaneously in working memory, and consequently reducing intrinsic cognitive load (Gerjets, Scheiter et al., 2006). They hypothesised that providing instructional explanations (prompting self-explanations) with molar (modular) examples improves learning performance. However, the results of their experiment indicated that neither intervention (instructional explanations and self-explanation prompts) improved learning performance. In fact, providing modular examples with self-explanation prompts led to a performance decrement because learners were forced to generate explanations of learning materials that had already been sufficiently understood.

In the domain of troubleshooting electrical circuits, van Gog, Paas et al. (2006) investigated the effects on transfer of process-oriented worked-examples (i.e. process information of the “why” and “how” added to worked-examples) and product-oriented worked-examples (i.e. standard worked-examples). The former can be argued to resemble worked-examples containing instructional explanations. van Gog, Paas et al. (2006) hypothesised that process-oriented worked-examples would increase investment of effort during training thus enhancing transfer. The results of their experiment partially confirmed this. While process-oriented worked-examples indeed resulted in an increased effort during training, their expected effects on transfer were not supported – that is, there was lower transfer performance in the process-oriented examples than in the product-oriented examples, thus suggesting an expertise reversal effect. Continuing from this study, van Gog, Paas et al. (2008) studied the effects of

presenting sequences of process-oriented and product-oriented worked-examples. They hypothesised that studying process-oriented worked-examples would initially lead to higher efficiency of transfer than studying product-oriented worked-examples. However this may become redundant if the same format continued as the training progresses because it would hinder learning. On the one hand, removing process information once learners have gained sufficient understanding by providing product-oriented worked-examples would continue to enhance learning. The results of their experiment confirmed the hypothesis and supported the expertise reversal effect found in van Gog, Paas et al.'s (2006) previous study. According to van Gog, Paas et al.'s (2008), expertise reversal effects might explain the failure to find positive effects on learning and transfer in previous study (e.g. Gerjets, Scheiter et al. 2006).

Berthold, Röder et al. (2011) tested the effects of conceptually-oriented explanation prompts on the learning process and explored the influence of such prompts on conceptual and procedural knowledge. The results of their experiment led to double-edged prompt effects – that is, positive effects on conceptual knowledge (mediated by principle-based understanding) and negative effects on procedural knowledge (indicated by problem-solving performance). The double-edged prompt effects might have been triggered by the complexity of the e-learning module on tax law used, which was characterised by a high level of intrinsic load. As Berthold, Röder et al. (2011 p. 74) argued “...learners seemed to have reached their upper limit of their working memory capacities by the prompts-induced focus on central principles and concepts (i.e., conceptual aspects) so that processing procedural aspects was hindered”. These results were supported by the findings obtained in the Berthold and Renkl's (2009) study when high-school learners learned probability. While the studies of Berthold, Röder et al. (2011) and Berthold and Renkl (2009) revealed double-edged prompt effects, an

earlier study (see Berthold, Eysink et al., 2009) revealed the opposite, namely that the prompts fostered both conceptual as well as procedural knowledge for the high prior knowledge learners of psychology who were learning about probability. They noted that the prior knowledge of the learners, the content of the prompts, and the learning tasks themselves were different in these studies and might have contributed to the inconsistent findings.

In accordance with Berthold, Eysink et al.'s (2009) finding, our research found positive effects of the plan-focused prompts on learners using the Structure-emphasising strategy. That is to say, the plan-focused prompts fostered procedural understanding for the high prior knowledge learners. This was indicated by significant difference in *overall* post-test scores between the high and low prior knowledge learners, with a large effect size. Moreover, the pre-test scores were systematically correlated with the *overall* post-test scores, as indicated by a large, positive correlation at the $p < .01$ level for both learners with high and low prior knowledge. Based on this evidence, it may be argued that the positive effects of the plan-focused prompts did not seem to occur for the low prior knowledge learners. In the light of the current formulation of cognitive load theory, learners with high prior knowledge had sufficient cognitive resources at their disposal to process interacting elements in working memory and such processing was possible within their levels of knowledge or expertise. In accord with Berthold, Röder et al.'s (2011) claim, learners with low prior knowledge could not benefit from the plan-focused prompts, presumably because they were induced on the plan-focused prompts that may have placed high demands on their working memory capacity thus leaving little capacity to deal with interacting elements germane to the learning task. It should be noted that the plan-focused prompts used in our study resembled the conceptually-oriented explanation prompts in Berthold, Röder et al.'s (2011) study. Whereas the emphasis

of the conceptually-oriented prompts was on explaining conceptual aspects (rationale) of the solution procedure, the plan-focused prompts largely focused on explaining procedural aspects of the program solution. For example, in our study, learners were prompted by a question such as “The value of flag is initialized to TRUE. Explain the condition under which the flag remains true”.

In conclusion, the interpretation of a *partial reversal* effect must remain tentative. It is not possible to substantiate this assumption based on the data obtained due to the different experimental procedures. Nevertheless the explanation seems plausible. To prove that *partial reversal* (Kalyuga & Renkl, 2010) can indeed occur, a further experiment is needed using repeated measures and following an experimental procedure that particularly investigates the expertise reversal effect and utilises the same set of instruments as used in this study.

8.8 Conclusion

On the evidence of the findings of this study, presenting learners with worked-examples using the Completion strategy proved to be beneficial for learning programming. However, that instruction should be tailored to the learners’ increasing knowledge levels (Kalyuga & Renkl, 2010). This can be achieved by means of a fading technique as introduced by Renkl, Atkinson et al. (2004). Recently Salden, Aleven, Schwonke and Renkl (2010) extended the work on fading by implementing an adaptive fading technique within a Cognitive Tutor. The results of their experiments showed that fading the worked-examples to adapt to a student’s current knowledge level led to improved learning than a fixed schedule of fading. Additionally, our reported findings demonstrated a *partial reversal* effect (see Kalyuga & Renkl, 2010) in the Completion strategy and have extended the findings of previous research

on expertise reversal effects. Finally, whereas van Merriënboer's (1990a) findings indicated little support for the idea that the negative effects of impulsivity (a feature of active learners) could be compensated by program completion, trends found in the present study seemed to suggest otherwise. The Completion strategy used in this study seemed suited to both the active and the reflective learners who seemed to benefit equally from the strategy.

Future work is needed to unravel some of the results that were not conclusive, especially in the findings of the Paired-method strategy. First, we hypothesised that the Paired-method strategy should be more efficient, (in terms of the learning *process* and *outcomes*) than the Completion strategy and the Structure-emphasising strategy. Unfortunately, we did not find the expected benefits of the Paired-method strategy. On the one hand, it may be argued that providing learners with the Paired-method strategy is still practical. An important avenue for future work is to investigate the effects of new *pairing sequence*, that is, to introduce learners to a pair of worked-examples using the Structure-emphasising format and subsequently a pair of worked-examples using the Completion format. Not only will this sequence ensure a more systematic transition from the Structure-emphasising to the Completion format, but learners' motivation would be maintained and this would enhance learning. This suggestion is in accordance with van Gog, Paas et al.'s (2008) argument, as discussed in Section 8.7. Also, it would be interesting to investigate the effects of prior knowledge via repeated measures experimental procedure. For the purpose of investigating cognitive overload, it would be worthwhile to adopt a dual task methodology. A questionnaire on the subjective measure also needs to be administered in order to map with results obtained using secondary task performance.

Second, we hypothesised that active learners would engage well with the Paired-method strategy and, conversely, would do worse than with either of the Completion or the Structure-emphasising strategy on their own. However, it was found that the active learners in the Paired-method strategy performed far lower than their active counterparts in the other two strategies and this contradicted one of the research hypotheses. Hence, another avenue for future work is to investigate the effects of the new *pairing sequence* on learners' motivation and learning, particularly on the active learners. The work needs to be conducted within a larger sample size, with adequate number of active and reflective learners. In summary, it would be interesting to investigate the effects of the new *pairing sequence*, in comparison to, say, the Completion strategy or in combination with prior knowledge or learners' learning style.

The explanation of procedural knowledge proved to be effective for high prior knowledge learners, but certainly not for low prior knowledge learners. One way to compensate for this negative effect on learners with low prior knowledge is to introduce conceptual-based explanations as introduced by Berthold, Röder et al. (2011). Conceptual-based explanation prompts, such as "Why does the program effectively compute the average of the numbers read in?" - excerpted from Soloway (1986), help to promote conceptual understanding and accordingly foster procedural knowledge (Berthold, Röder et al., 2011). The conceptual-based explanation prompt is more easily constructed and hence less difficult than procedural-based explanations. Consequently, learners have more cognitive resources at their disposal to invest effort in understanding the learning materials. In this way, both learners with low prior knowledge as well those with high prior knowledge would potentially benefit from the explanation exercise. The explanation effect as revealed in our study needs to be further

investigated in future studies, within the same domain. Besides, it would be interesting to investigate the explanation effect in two different contexts: procedural and conceptual before any recommendations are made. One final point to note is that the reflective learners in the Structure-emphasising strategy performed far lower than their reflective counterparts in the other two strategies, which is contrary to our early predictions and thus needs further experimental confirmation.

Looking at the effects on learning styles, the results did not reach significance, although effects ranged from a medium to a large effect size. It may be argued that learning style may influence the effects of learners' cognitive load. However, this suggestion is only applicable to this sample study. Further research should be conducted with a larger sample size to corroborate this claim.

As a final point, learners with low prior knowledge did not seem to benefit from learning with any of the three worked-example strategies. A possible explanation for this may be the high element interactivity materials (i.e. counter, action in the body of the loops) that needed to be processed simultaneously within the learner's limited working memory capacity. Thus one question remains open: how can one design worked-examples that may help learners with low prior knowledge deal with complex materials, such as learning loops?

Chapter 9 Conclusions

This chapter summarises the theoretical and empirical contributions of the present research. It reflects on the limitations of the research and offers an agenda of future work.

9.1 Contributions of the thesis

This thesis makes a contribution to CLT, in that it has explored a factor which has received insufficient attention and may influence learning outcomes. More specifically, we developed and explored the idea that learning styles may interact with learners' cognitive load, and thus affect learning outcomes. However, this conclusion must remain tentative until further experimental confirmation has been made.

The thesis has elucidated some of the practical issues concerning the reliability and sensitivity of cognitive load measures which have been debated among researchers in the field of CLT. Self-report ratings of cognitive load measures proved highly subjective and, in agreement with de Jong (2010), such subjective measures are not always sensitive to experimental manipulations.

The thesis has also explored the conceptual difficulties with learning efficiency measures that may lead to an unclear experimental situation (de Jong, 2010; Moreno & Valdez, 2005). In our study, for example, it was difficult to interpret the efficiency (in terms of learning outcomes) of the Structure-emphasising strategy in relation to the efficiency of the Paired-method strategy. In other words, the former strategy showed a high level of performance

combined with a high level of invested mental effort whereas the latter strategy showed a low level of performance combined with a low level of invested mental effort. Thus, further work on this issue is clearly called for.

This thesis has extended the findings of previous research on the completion effect. That is, on the evidence of our findings, presenting learners with worked-examples using the Completion strategy proved to be beneficial for learning programming. However, that instruction should be tailored to the learners' increasing knowledge levels (Kalyuga & Renkl, 2010). Moreover, trends found in our study suggest that the Completion strategy suits both the active and reflective learners as they both seemed to benefit equally from the strategy. Thus was at variance with the findings of van Merriënboer (1990a), and so merits further work.

The thesis has also advanced our understanding of the *partial reversal* effect in the Completion strategy and the explanation effect in the Structure-emphasising strategy. However, our interpretation of *partial reversal* effect must remain tentative until further empirical work has been undertaken (particularly using a repeated measures methodology).

The thesis has introduced and evaluated the Paired-method strategy but did not find the expected benefits. One way of overcoming an issue with the *pairing sequence* may be to introduce learners to it via a pair of worked-examples using the Structure-emphasising format and subsequently a pair of worked-examples using the Completion.

The work has developed and evaluated the use of a learning environment, LECSES, for learning from worked examples using three learning strategies. The environment has an associated editing suite that allows for the inclusion of new worked-examples relatively easily.

Additionally, LECSES introduced two unique features, first the explanation guidance through plan-focused prompt and hints. Second, the creation of a modification exercise through the use of advanced web techniques (as discussed in Chapter 4, Section 4.3).

Finally, the work has built on the present research in the area of learning styles, psychology of programming, especially within the education literature, and from worked-example research to CLT and WMC research.

9.2 Research limitations

This research is not without its limitations and these will be discussed in the following section.

9.2.1 Sample size

The sample size of this study was small ($n = 110$) with an unequal number of Active, Balanced, and Reflective participants in each experimental group (e.g. 9, 23, and 5 respectively for the Structure-emphasising group). However, the pattern of these numbers was more or less similar across the three strategy groups. One noteworthy point is that these numbers reflect the general population (see Chapter 3 for more details). For this reason, we

could not easily get a large and equal number of active and reflective learners taking part in the experiment. Due to the small sample size, many of the results did not reach significance, particularly when comparing differential effects between the learning styles within each strategy group. We were also unable to balance the degree of programming prior knowledge across the three groups, and it is possible that this skewed the results slightly.

9.2.2 WMC and ILS instruments

We were not able not replicate the findings Graf, Liu et al. (2009) with respect to a relationship between learning style and working memory capacity. This could be due to several factors as discussed in the subsequent paragraphs.

In our study, two versions (pen and paper based) of the ILS questionnaire (Felder & Soloman, n.d.) were administered to the learners. These were the English version (an original version of the ILS) and the Malay version. The English version was administered to the International students and that the Malay version was administered to the local students. In the study conducted by Graf, Liu et al. (2009), three questions from the ILS questionnaire were removed due to reliability reasons and one question was removed for the active/reflective dimension. Clearly, the ILS instrument used in Graf, Liu et al.'s (2009) study was different from the one that was administered to our participants, in addition to the Malay version already mentioned.

In our study, the correlation coefficient between the WMC values and the ILS values was measured using the whole continuum of the active/reflective dimension. In the Graf, Liu et al.'s (2009) study, their analysis of correlation was performed on a sub-dataset. Their dataset

was split into two sub-datasets, distinguishing the subset of active/balanced from the subset of reflective/balanced. The active/balanced subset included those with a preference for either an active or balanced style, whose score was smaller than or equal to -3 on the ILS active/reflective dimension. In contrast, the reflective/balanced subset included those with a preference for either a reflective or balanced style, whose score is greater than or equal to 3.

As described in Chapter 5, the measures generated by the online version of Web-OSPAN had to be written down manually by the learners and were not recorded automatically. Therefore, it is difficult to be certain that these scores were recorded accurately. Moreover, the only measures available from Web-OSPAN were the number of correct arithmetic operations, the total number of words correctly recalled, and the word span in working memory. Without the mean response latency and a partial correct memory span¹, there must remain some uncertainty in the conclusions. Finally, it is unclear how the previous study (Graf, Liu et al., 2009) drew the boundary between the low WMC group and the high WMC group. As for our study, values greater than or equal to 30 indicated high WMC whereas values smaller than 30 indicated low WMC.

9.2.3 LECSES

LECSES ran on a shared host and the system could handle up to 100 users at one time with reasonable performance. In fact, the number of participants who took part in the experiment (during the learning phase) slightly exceeded the capacity. The consequent server overload caused some requested pages not to respond or load quickly enough. Nonetheless, the

¹ An extra measure specially added to Web-OSPAN that counts correct words regardless of the order of words (0-60).

problem only occurred occasionally. Note that the timer (that recorded the amount of time taken by a participant to complete an exercise) started once the requested page had been successfully loaded. The timer kept running even if the page suddenly stopped responding. For a similar reason, several items of data on perceived effort and difficulty had to be recorded manually while some of these data were simply missing.

On the client-side, LECSES used a complex JavaScript (particularly with regard to the Completion format) that needed to be processed quickly on the client's (i.e. participant) machine. The lack of performance of some of the clients' machines caused the script not to run properly so that the participant might have had to re-do an exercise within the remaining time given. This fact came to light from an interview after the experiment with one of the participants. Despite these problems, the system response time was generally acceptable, as indicated in the replies to the questionnaire.

Due to an error, there were two lines of program code missing for the explanation exercise in one problem (Problem 3 of Set 2). These lines were mistakenly 'hidden', i.e. the lines were 'highlighted' in green during the course of composing reflection exercise. Note that the same source of material (i.e. Problem 3) was used for composing the explanation and reflection exercises via the same editor (see Figure 4.14 for more details). For a similar reason, the reflection exercise (of Set 3) using the Structure-emphasising format was unintentionally hybrid. For this reflection exercise, the participants were no longer asked to predict the program's behaviour with respect to the line(s) of code highlighted in green, instead they were asked to think about what additions and deletions that the program needed to ensure it solved the modified problem.

In addition, we identified two inappropriate pieces of feedback associated with the Completion format. First, even though a correct answer was given, LECSES failed to ‘recognise’ this. Again, this has to do with the under-performance of the client’s machine as described earlier but again this only occurred on a few machines. Second, there were a limited number of pre-defined, possible correct answers for a text box ‘embedded’ within a program solution (see Chapter 4, for further explanation on this) and as a result, a correct answer given was regarded as wrong answer.

It is possible that the LECSES learning environment was in general more suited for reflective learners, in that it did not support active and dynamic learning. However, if the two learning styles are to be compared in a 2-by-2 factorial design study, the differential effects on learning and transfer may be readily observed. Another noteworthy point is the possible impact of time pressure (with the on-screen ticking clock) which may possibly have caused learners to act differently from their usual learning styles.

9.2.4 Other limitations

Despite being assured by an IT officer, a power cut happened during the first half hour of the main experiment. This was during the warm-up session that let participants familiarise themselves with LECSES. The experiment had to be discontinued until the power was restored. The timer was reset and old answers were flushed-out at the point where the interruption occurred (i.e. the exercise the participants had been currently working with). Participants resumed from where they had stopped. Moreover, due to power cut issue that lasted nearly two hours, the time allocated had to be reduced to 15 minutes for each exercise.

As a final point, participants in the Structure-emphasising group were given, inadvertently, a slight hint about the nature of the transfer tests, as the lecturer (who helped out during the experiment) wanted to motivate them to take part in the experiment more actively.

9.3 Future work

We suggest a number of future lines of research. Graf, Liu et al. (2009) found a significant relationship between learning style and working memory capacity. In contrast, this finding was not replicated in our study. Nevertheless, we argue that investigating this relationship, and hence the differential effects of cognitive load on active and reflective learners appears to be a promising area for further investigation and should be taken into account when designing CLT experiments. It is also worthwhile to consider using alternative learning style inventories (among others, Kolb's Learning Style Inventory²; Honey and Mumford's Learning Styles Questionnaire³; The Myers-Briggs Type Indicator⁴), different ways of measuring working memory capacity, and more importantly, working with a larger sample size so as to substantiate the tentative hypothesis (as mentioned in section 9.2).

Learners in the Paired-method strategy were considerably affected by the pairing sequence (i.e. SE-CS, SE-CS, and so on). As a new pairing sequence is proposed, a further experiment is needed to investigate its effects on learners' motivation and learning, particularly on the active learners. A possible experimental design was previously discussed in Chapter 8, in Section 8.8 -- paragraph 2 and 3.

² The Kolb Learning Style Inventory, Version 3 (Kolb, 1999).

³ The learning styles helper's guide (Honey & Mumford, 2000).

⁴ Manual: a guide to the development and use of the Myers-Briggs Type Indicator (Myers & McCaulley, 1998).

Reflective learners in the Structure-emphasising strategy performed far worse than their reflective counterparts in the other two strategies, which was contrary to our prediction. Thus the question arises as to whether reflective learners in the Structure-emphasising strategy may perhaps have exhibited a preference for an active style. Hence, more studies are clearly called for to explore this finding.

It has been found that the explanation of procedural knowledge proved to be effective for high prior knowledge learners, but certainly not for low prior knowledge learners. Thus, it would be interesting to investigate this explanation effect for two different contexts, i.e. procedural vs. conceptual explanation using repeated measures on 2 by 2 factorial design before any practical recommendation is made.

As a final point, low prior knowledge learners did not seem to benefit particularly from learning with any of the three worked-example strategies. Thus one question remains open. How can one design worked-examples and a worked example strategy that helps low prior knowledge learners deal with complex materials, such as learning loops?

APPENDIX A

Participant consent form

Thank you for agreeing to be a participant in this scientific study. Your participation is very much appreciated. I hope that participation will be of value to you in that I hope that you will learn/revise some Java concepts as a result.

Aim

This experiment is investigating how different worked-example strategies differentially affect students learning programming, taking into account individual learning style. The focus of the experiment is on the topic of loops.

Phases and activities of the experiment

The experiment consists of three phases:

Phase 1 starts by asking you to complete a questionnaire about your programming background. You will also be asked to complete the Felder-Soloman Index of Learning Styles questionnaire¹. The questionnaire helps to determine your preferred learning styles. In this phase, you will also be asked to do the operation word span task (OSPAN)². This task helps to determine your working memory capacity. Finally, you will be asked to sit for a pre-test. The pre-test assesses your prior knowledge on a range of programming topics, including loops.

In Phase 2, you will work with 3 sets of 2 problems (6 problems altogether) using LECSES, a web-based worked-example system. After working with each problem, you will be asked to give an estimate of the difficulty of the learning method and of the degree to which you have learned new Java concepts from the materials.

Phase 3 concerned with a program development and coding test and you will be requested to solve 4 programming problems. Please note that the marks obtained from this test will be treated as your mid-term test. After each problem, you will be requested to give an estimate of the mental effort invested in solving the problem. In this phase, you will also be asked to complete a questionnaire about the worked-example strategy and LECSES. Finally, an individual follow-up with a few selected participants will be conducted individually.

You will be informed of the activities and date/time in advance. Further instruction will also be given prior to the activities.

Statement of voluntary participation and confidentiality

Your participation is voluntary and you may withdraw from this experiment at any time. You will not be identified by name in any publications arising from the work and that the results on the experiment will be kept indefinitely and confidentially. Only the mid-term test results will be given to your course tutor. All the other data will remain confidential.

Statement of consent

I have read and understand the above information and hereby consent to participate in this experiment.

Name:

Matric no:

Signature:

Date:

¹ Index of Learning Styles (Felder & Soloman), <http://www.engr.ncsu.edu/learningstyles/ilsweb.html>

² OSPAN (Turner & Engle, 1989)

APPENDIX B

Participant programming background

Matric no: _____

- 1 Have you taken any programming courses before? (circle) YES / NO

If YES, answer question 2, 3, 4.

If NO, answer question 3 only.

- 2 State the programming language(s) that you have learnt from the list below (circle).

C / C++ / C# / Basic / Visual Basic / Pascal / Java / others: _____

- 3 You are asked to solve a simple problem to compute the summation of two even numbers.
List the steps to solve the problem.

- 4 Use a programming language you have learnt to write a program to solve the problem in question 3.

Index of Learning Styles Questionnaire

**Barbara A. Soloman
First-Year College
North Carolina State University
Raleigh, North Carolina 27695**

**Richard M. Felder
Department of Chemical Engineering
North Carolina State University
Raleigh, NC 27695-7905**

Directions

Please provide us with your full name. Your name will be printed on the information that is returned to you.

Full Name _____

For each of the 44 questions below select either "a" or "b" to indicate your answer. Please choose only one answer for each question. If both "a" and "b" seem to apply to you, choose the one that applies more frequently. When you are finished selecting answers to each question please select the submit button at the end of the form.

1. I understand something better after I
 - ☐ (a) try it out.
 - ☐ (b) think it through.
2. I would rather be considered
 - ☐ (a) realistic.
 - ☐ (b) innovative.
3. When I think about what I did yesterday, I am most likely to get
 - ☐ (a) a picture.
 - ☐ (b) words.

APPENDIX C

4. I tend to
 - ☐ (a) understand details of a subject but may be fuzzy about its overall structure.
 - ☐ (b) understand the overall structure but may be fuzzy about details.
5. When I am learning something new, it helps me to
 - ☐ (a) talk about it.
 - ☐ (b) think about it.
6. If I were a teacher, I would rather teach a course
 - ☐ (a) that deals with facts and real life situations.
 - ☐ (b) that deals with ideas and theories.
7. I prefer to get new information in
 - ☐ (a) pictures, diagrams, graphs, or maps.
 - ☐ (b) written directions or verbal information.
8. Once I understand
 - ☐ (a) all the parts, I understand the whole thing.
 - ☐ (b) the whole thing, I see how the parts fit.
9. In a study group working on difficult material, I am more likely to
 - ☐ (a) jump in and contribute ideas.
 - ☐ (b) sit back and listen.
10. I find it easier
 - ☐ (a) to learn facts.
 - ☐ (b) to learn concepts.
11. In a book with lots of pictures and charts, I am likely to
 - ☐ (a) look over the pictures and charts carefully.
 - ☐ (b) focus on the written text.
12. When I solve math problems
 - ☐ (a) I usually work my way to the solutions one step at a time.
 - ☐ (b) I often just see the solutions but then have to struggle to figure out the steps to get to them.
13. In classes I have taken
 - ☐ (a) I have usually gotten to know many of the students.
 - ☐ (b) I have rarely gotten to know many of the students.
14. In reading nonfiction, I prefer
 - ☐ (a) something that teaches me new facts or tells me how to do something.
 - ☐ (b) something that gives me new ideas to think about.

APPENDIX C

15. I like teachers
 - ☐ (a) who put a lot of diagrams on the board.
 - ☐ (b) who spend a lot of time explaining.
16. When I'm analyzing a story or a novel
 - ☐ (a) I think of the incidents and try to put them together to figure out the themes.
 - ☐ (b) I just know what the themes are when I finish reading and then I have to go back and find the incidents that demonstrate them.
17. When I start a homework problem, I am more likely to
 - ☐ (a) start working on the solution immediately.
 - ☐ (b) try to fully understand the problem first.
18. I prefer the idea of
 - ☐ (a) certainty.
 - ☐ (b) theory.
19. I remember best
 - ☐ (a) what I see.
 - ☐ (b) what I hear.
20. It is more important to me that an instructor
 - ☐ (a) lay out the material in clear sequential steps.
 - ☐ (b) give me an overall picture and relate the material to other subjects.
21. I prefer to study
 - ☐ (a) in a study group.
 - ☐ (b) alone.
22. I am more likely to be considered
 - ☐ (a) careful about the details of my work.
 - ☐ (b) creative about how to do my work.
23. When I get directions to a new place, I prefer
 - ☐ (a) a map.
 - ☐ (b) written instructions.
24. I learn
 - ☐ (a) at a fairly regular pace. If I study hard, I'll "get it."
 - ☐ (b) in fits and starts. I'll be totally confused and then suddenly it all "clicks."
25. I would rather first
 - ☐ (a) try things out.
 - ☐ (b) think about how I'm going to do it.

APPENDIX C

26. When I am reading for enjoyment, I like writers to
 - ☐ (a) clearly say what they mean.
 - ☐ (b) say things in creative, interesting ways.
27. When I see a diagram or sketch in class, I am most likely to remember
 - ☐ (a) the picture.
 - ☐ (b) what the instructor said about it.
28. When considering a body of information, I am more likely to
 - ☐ (a) focus on details and miss the big picture.
 - ☐ (b) try to understand the big picture before getting into the details.
29. I more easily remember
 - ☐ (a) something I have done.
 - ☐ (b) something I have thought a lot about.
30. When I have to perform a task, I prefer to
 - ☐ (a) master one way of doing it.
 - ☐ (b) come up with new ways of doing it.
31. When someone is showing me data, I prefer
 - ☐ (a) charts or graphs.
 - ☐ (b) text summarizing the results.
32. When writing a paper, I am more likely to
 - ☐ (a) work on (think about or write) the beginning of the paper and progress forward.
 - ☐ (b) work on (think about or write) different parts of the paper and then order them.
33. When I have to work on a group project, I first want to
 - ☐ (a) have "group brainstorming" where everyone contributes ideas.
 - ☐ (b) brainstorm individually and then come together as a group to compare ideas.
34. I consider it higher praise to call someone
 - ☐ (a) sensible.
 - ☐ (b) imaginative.
35. When I meet people at a party, I am more likely to remember
 - ☐ (a) what they looked like.
 - ☐ (b) what they said about themselves.
36. When I am learning a new subject, I prefer to
 - ☐ (a) stay focused on that subject, learning as much about it as I can.
 - ☐ (b) try to make connections between that subject and related subjects.

APPENDIX C

37. I am more likely to be considered
- ☐ (a) outgoing.
 - ☐ (b) reserved.
38. I prefer courses that emphasize
- ☐ (a) concrete material (facts, data).
 - ☐ (b) abstract material (concepts, theories).
39. For entertainment, I would rather
- ☐ (a) watch television.
 - ☐ (b) read a book.
40. Some teachers start their lectures with an outline of what they will cover. Such outlines are
- ☐ (a) somewhat helpful to me.
 - ☐ (b) very helpful to me.
41. The idea of doing homework in groups, with one grade for the entire group,
- ☐ (a) appeals to me.
 - ☐ (b) does not appeal to me.
42. When I am doing long calculations,
- ☐ (a) I tend to repeat all my steps and check my work carefully.
 - ☐ (b) I find checking my work tiresome and have to force myself to do it.
43. I tend to picture places I have been
- ☐ (a) easily and fairly accurately.
 - ☐ (b) with difficulty and without much detail.
44. When solving problems in a group, I would be more likely to
- ☐ (a) think of the steps in the solution process.
 - ☐ (b) think of possible consequences or applications of the solution in a wide range of areas.

When you have completed filling out the above form please click on the Submit button below. Your results will be returned to you. If you are not satisfied with your answers above please click on Reset to clear the form.

Soal selidik Index of Learning Styles

Barbara A. Soloman
First-Year College
North Carolina State University
Raleigh, North Carolina 27695

Richard M. Felder
Department of Chemical Engineering
North Carolina State University
Raleigh, NC 27695-7905

Nombor matrik:

Arahan

Bagi soalan-soalan di bawah, sila bulatkan jawapan anda (samada a atau b). Jika kedua-dua pilihan adalah benar bagi anda, pilihlah jawapan yang lebih kerap benar.

1. Saya lebih memahami sesuatu selepas saya
 - a. mencubanya dahulu.
 - b. berfikir tentang perkara tersebut.
2. Saya lebih suka dianggap
 - a. realistik.
 - b. inovatif.
3. Apabila saya memikirkan tentang apa yang telah saya lakukan semalam, saya akan terbayang
 - a. suatu gambaran.
 - b. perkataan.
4. Saya lebih cenderung untuk
 - a. memahami ciri terperinci suatu perkara tetapi samar-samar tentang struktur keseluruhan.
 - b. memahami struktur keseluruhan tetapi samar-samar tentang ciri terperinci.
5. Apabila saya sedang mempelajari suatu yang baru, yang akan membantu saya ialah
 - a. bercakap tentang perkara tersebut.
 - b. berfikir tentang perkara tersebut.
6. Jika saya seorang guru, saya lebih suka mengajar kursus
 - a. yang melibatkan fakta dan situasi kehidupan sebenar.
 - b. yang melibatkan idea dan teori.
7. Saya lebih suka mendapatkan maklumat baru dalam bentuk
 - a. gambar, gambarajah, graf, atau peta.
 - b. arahan bertulis atau maklumat lisan.
8. Setelah saya memahami
 - a. semua bahagian, saya akan memahami perkara sepenuhnya.
 - b. perkara sepenuhnya, saya akan memahami bagaimana bahagian-bahagian berkaitan antara satu sama lain.

APPENDIX D

9. Semasa sebuah kumpulan belajar sedang membincangkan bahan sukar, saya lebih cenderung
 - a. turut serta dan menyumbangkan idea.
 - b. duduk diam dan mendengar sahaja.
10. Lebih mudah bagi saya untuk
 - a. mempelajari fakta.
 - b. mempelajari konsep.
11. Dalam sebuah buku yang mengandungi banyak gambar dan carta, saya lebih cenderung untuk
 - a. melihat gambar dan carta dengan lebih teliti.
 - b. menumpukan perhatian kepada tulisan.
12. Apabila saya menyelesaikan masalah matematik
 - a. saya mendapatkan penyelesaian langkah demi langkah.
 - b. saya selalu dapat bayangkan penyelesaiannya tetapi menghadapi masalah untuk mendapatkan langkah-langkah sebelum penyelesaian tersebut.
13. Dalam kelas, saya
 - a. telah mengenali kebanyakan pelajar-pelajar.
 - b. hanya mengenali segelintir daripada pelajar-pelajar.
14. Jika membaca buku bukan fiksi, saya lebih gemar
 - a. bahan bacaan yang mengajar fakta baru atau memberitahu cara-cara untuk melakukan sesuatu.
 - b. bahan bacaan yang memaparkan idea baru untuk difikirkan.
15. Saya suka guru yang
 - a. mengajar menggunakan banyak gambarajah.
 - b. mengajar dengan memberi penerangan yang panjang lebar.
16. Jika saya mengkaji sebuah cerpen atau novel,
 - a. saya cuba mengaitkan plot-plot cerpen/novel tersebut untuk mencari tema cerpen/novel itu.
 - b. saya hanya dapat gambaran tentang tema cerpen/novel tersebut setelah habis membaca, kemudian baru mengaitkannya dengan plot-plot cerpen/novel itu.
17. Semasa mula menyelesaikan soalan tugas, besar kemungkinan saya akan
 - a. segera mencari jalan penyelesaian masalah tersebut.
 - b. cuba memahami soalan tersebut dengan sepenuhnya, kemudian baru menyelesaikannya.
18. Saya lebih suka kepada idea yang berunsur
 - a. kepastian.
 - b. teori.
19. Saya paling boleh mengingat
 - a. apa yang dilihat.
 - b. apa yang didengar.
20. Bagi saya, lebih penting jika seseorang pensyarah
 - a. mengajar bahan mengikut langkah yang berturutan.
 - b. memberi gambaran menyeluruh tentang apa yang hendak diajar dan mengaitkannya dengan subjek-subjek lain.
21. Saya lebih gemar belajar
 - a. secara berkumpulan.
 - b. sendirian.

APPENDIX D

22. Saya boleh dikatakan
 - a. berhati-hati tentang butiran terperinci dalam sesuatu tugas.
 - b. kreatif tentang cara saya menyampaikan sesuatu tugas.
23. Untuk sampai ke suatu destinasi yang baru, saya lebih suka
 - a. menggunakan peta.
 - b. menggunakan arahan bertulis.
24. Proses pembelajaran saya
 - a. dengan kadar sederhana. Sekiranya saya tekun berusaha, tentu saya akan dapat memahami pelajaran saya dengan sepenuhnya.
 - b. agak berkecamuk dan memeningkan tetapi sampai di satu waktu, saya akan dapat memahami kesemuanya.
25. Saya lebih suka
 - a. mencuba sesuatu terlebih dahulu.
 - b. memikirkan cara untuk melakukan sesuatu terlebih dahulu.
26. Jika saya membaca, saya lebih meminati penulis-penulis yang
 - a. menyampaikan karya mereka dalam bahasa yang jelas dan mudah difahami.
 - b. menulis dengan cara yang kreatif dan menarik.
27. Sekiranya saya memerhati gambarajah atau lakaran dalam kelas, saya akan lebih mengingati
 - a. gambarajah tersebut.
 - b. apa yang dikatakan tentang gambarajah tersebut oleh pensyarah.
28. Dalam proses memahami sesuatu maklumat baru, besar kemungkinan saya akan
 - a. memberi perhatian kepada butiran terperinci dan lupa tentang gambaran keseluruhan maklumat tersebut.
 - b. cuba memahami maklumat tersebut secara keseluruhan sebelum memberi perhatian kepada butiran terperinci.
29. Saya lebih mudah mengingati
 - a. sesuatu yang telah saya lakukan.
 - b. sesuatu yang telah saya fikirkan.
30. Sekiranya saya diberi tugas untuk dibuat, saya akan
 - a. memikirkan satu cara untuk menyiapkannya.
 - b. memikirkan pelbagai cara baru untuk menyiapkannya.
31. Sekiranya seseorang itu menunjukkan saya data, saya lebih suka
 - a. carta atau graf.
 - b. rumusan bertulis tentang data tersebut.
32. Semasa menulis kertas kerja, besar kemungkinan saya akan
 - a. memikirkan atau menulis bermula dari awal dan meneruskan hingga ke akhir.
 - b. memikirkan atau menulis bahagian yang berbeza kemudian baru menyusunnya.
33. Sekiranya saya perlu menyiapkan kerja berkumpulan, saya akan
 - a. mengadakan sesi 'brain storming' (mengeluarkan idea) dengan semua ahli sekaligus.
 - b. 'brain storm' secara individu, kemudian baru membentangkannya kepada ahli kumpulan yang lain.
34. Pujian tinggi yang saya berikan kepada seseorang ialah dengan melabelnya
 - a. arif dan rasional.
 - b. mempunyai daya imaginasi yang tinggi.

APPENDIX D

35. Sekiranya saya berjumpa dengan orang baru, besar kemungkinan saya akan mengingat
a. wajah mereka.
b. perkara yang diperkatakan oleh mereka mengenai diri mereka sendiri.
36. Sewaktu mempelajari subjek yang baru, saya lebih suka
a. fokus sepenuhnya kepada subjek tersebut, dan mempelajarinya dengan sebanyak yang mungkin.
b. cuba mengaitkan subjek tersebut dengan subjek-subjek yang lain.
37. Saya lebih dikenali sebagai seorang yang
a. 'outgoing', suka berinteraksi dengan orang lain.
b. 'reserved', lebih suka menyendiri.
38. Saya lebih meminati kursus-kursus yang memberi penekanan kepada
a. bahan pembelajaran yang konkrit (fakta dan data).
b. bahan pembelajaran yang abstrak (konsep dan teori).
39. Untuk hiburan, saya lebih gemar
a. menonton televisyen.
b. membaca buku.
40. Sesetengah pensyarah memulakan pelajaran dengan memberi sedikit gambaran atau 'outline' tentang apa yang akan diajar. Bagi saya, ini
a. sedikit sebanyak menolong saya dalam memahami apa yang ingin diajar nanti.
b. sangat berguna dalam memahami apa yang akan diajar.
41. 'Kerja berkumpulan dengan markah yang diberi adalah sama bagi setiap ahli dalam kumpulan tersebut'. Tugas sebegini
a. sangat saya sukai.
b. tidak begitu saya sukai.
42. Sekiranya saya membuat pengiraan panjang
a. saya akan mengulangi kesemua langkah-langkah dan menyemak semula kerja saya dengan teliti.
b. saya tidak suka menyemak semula kerana beranggapan itu agak membebankan dan mesti memaksa diri untuk melakukannya.
43. Saya dapat menggambarkan sesuatu tempat yang pernah dikunjungi
a. dengan mudah dan agak tepat.
b. dengan agak sukar dan tidak secara terperinci.
44. Dalam menyelesaikan masalah dalam sebuah kumpulan, selalunya saya
a. memikirkan cara-cara untuk menyelesaikan masalah tersebut.
b. memikirkan akibat/kegunaan bagi penyelesaian dalam pelbagai bidang.

APPENDIX E

Web Operation Word Span Task (Web-OSPAN)¹

Matric no: _____

Instructions:

1. Please write down your matric no.
2. Go to <http://wmc.lecses.com/login.php>
3. Check the checkbox.
4. Enter User ID and password.
5. Please read the instructions for operation word task CAREFULLY and relax.
6. DO NOT START THE TASK UNTIL YOU ARE TOLD TO DO SO.
7. At the end of the task, you will see your scores on a screen. Please write down your scores CORRECTLY and CLEARLY and completed time for the task.

You scored

Correct arithmetic operation : _____

Total word correctly recalled : _____

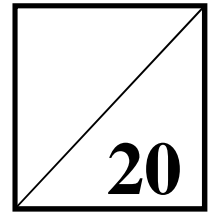
Word span in working memory : _____

Time completed : _____

PLEASE ENSURE THAT THE SCORES ARE WRITTEN DOWN CORRECTLY

¹ Web-OSPAN (Taiyu Lin), available at <http://kinshuk.athabascau.ca/webospan/>

Programming Pre-test



Matric no: _____

1. The following program supposedly calculates the area of a circle.¹

```
1  import java.util.Scanner;
2
3  public class AreaOfCircle{
4
5      public static void main(String args[]){
6
7          double radius;
8
9          Scanner keyboard = new Scanner(System.in);
10
11          System.out.print("Enter radius of the circle: ");
12          radius = keyboard.nextInt();
13
14          System.out.printf("The area of the circle is %.2f\n",
15                             radius*radius*3.14159);
16      }
17 }
```

Modify the above program with regard to the following:

- What will happen if the user enters 2.3 for radius? Rewrite the code to make the program work as it should.
- Make `PI` a named constant that holds the double value 3.14159.
- Declare a variable called `area` that holds the value for the area of the circle.
- Write a line of code that computes an area of the circle using `PI` and assigns this value to variable `area`.
- Rewrite line 14 that displays a message and the result of the area of the circle.

[5 marks]

Answer for question 1

- Input mismatched. `radius = keyboard.nextDouble();`
- `final double PI = 3.14159`
- `double area;`
- `area = radius * radius * PI;`
- `System.out.printf("The area of the circle is %.2f\n",area);`

¹ Question 1 and 2 were adapted from Absolute Java by Walter, J. Savitch, 4th Edition, Pearson Education

2. For question a and d, determine the value of C and the value of status when the code has finished execution. For question b and c, determine the value (true or false) of the expressions. Assume that the variables are declared and initialised as follows: ¹

```
A = 10;
B = 12;
```

C = 5

True

False

```
status = true
```

3. Write a switch statement that prints the name of a colour depending on the number entered.

1, 2, 3	Yellow
4, 5	Pink
Others	Green

276

APPENDIX F

Answer for question 3

```
switch(number){  
    case 1:  
    case 2:  
    case 3: System.out.println("The colour is Yellow");  
        break;  
    case 4:  
    case 5: System.out.println("The colour is Pink");  
        break;  
    default: System.out.println("The colour is Green");  
        break;  
}
```

4. Trace the output produced by the following program fragment.

```
int i, j;  
i = 1;  
  
while ( i <= 4 ){  
    j = 1;  
  
    while ( j <= i){  
        System.out.print(i*j);  
        j++;  
    }  
  
    System.out.print("\n");  
    i++;  
}
```

Answer for question 4

I	J	Output
1	1	1
2	1 2	2 2 4
3	1 2 3	3 3 6 3 6 9
4	1 2 3 4	4 4 8 4 8 12 4 8 12 16

[4 marks]

5. Write a program using while loop that accepts positive integers from the user and then computes the sum of the integers. The program finally displays the results of the summation together with the largest integer entered. The loop ends with the sentinel input.

[6 marks]

APPENDIX F

Answer for question 5

```
int sum, number, largest;
sum = 0;
largest = 0;

System.out.print("Enter positive integers: ");
number = keyboard.nextInt();

while (number != -1){

    sum = sum + number;

    if (number > largest)
        largest = number;

    System.out.print("Enter positive integers: ");
    number = keyboard.nextInt();
}

System.out.println("The sum is " + sum);
System.out.println("The largest number is " + largest);
```


APPENDIX G

Example problems for the learning phase

Average score

Zara is a freelance academic consultant who has been appointed by the principal of Permata Kolej to assess students' personalities based on two test scores, namely Test A and Test B. The personality tests were administered on 20 students. Zara had a program written for her that calculated and displayed the average of the two test scores for each student. Additionally, the program displayed at its end the overall average test score for Test B for female students who scored above 15. This example illustrates a program that prompts the user to input the student's gender and test scores (in the range of 0 to 20) obtained by each student. Assume that the user will enter a valid input.

```
Gender (F for Female / M for Male): M
Score for Test A: 12
Score for Test B: 15
The average of the two test scores was 13.50

Gender (F for Female / M for Male): F
Score for Test A: 12
Score for Test B: 16
The average of the two test scores was 14.00

Gender (F for Female / M for Male): F
Score for Test A: 15
Score for Test B: 17
The average of the two test scores was 16.00
:

Gender (F for Female / M for Male): F
Score for Test A: 10
Score for Test B: 12
The average of the two test scores was 11.00

The average score for Test B (score > 15) for female student was xxx
```

Prompts and description of each plan

Loop entry condition, counter-controlled plan

Explain how the expression `(counter < 20)` in the while statement is evaluated.

The expression is evaluated by comparing the value of `counter` with 20. The body of the loop continues to execute until the value of `counter` reaches 20.

Hint 1: Please refer to counter loop plan.

Hint 2: Condition under which the loop continues.

Running total loop plan

Explain how the running total variables are updated. The variables are updated when the condition `if (gender == 'F' && scoreB > 15)` evaluates to true. The loop adds the value of `female_scoreB` with the new value of `scoreB` and increments the value of `female_count` by 1.

APPENDIX G

Hint 1: Condition under which, the running total variables are updated with the current values.

Counter loop plan

The value of `counter` at which the loop starts is initialised to 0. Explain the reasoning behind this initialisation taking into account the expression `(counter < 20)` in the `while` statement. That initialisation makes that the expression `(counter < 20)` in the `while` statement true before any loop is executed. `Counter` ensures that each repetition makes progress towards the expression becoming false by increasing its value by one until it reaches 20.

Hint 1: Keep track the number of times the user has entered data.

Hint 2: `counter` will increment itself by 1 each time through the loop.

Guard and division plan

Explain what happens if the `while` loop ends with the value of `female_count` equal to 0. The condition `female_count != 0` becomes false, causing the program prints “No average computed”. Otherwise, an appropriate message with the value of average will be printed. Division by 0 causing run time error and will immediately stop the program.

Hint 1: Division by zero.

Modification / reflection exercise

Modify the above program so that the program additionally calculates and displays the overall average test score for Test A and Test B separately.

```
Gender (F for Female / M for Male): F
Score for Test A: 12
Score for Test B: 16
The average of the two test scores was 14.00
```

```
Gender (F for Female / M for Male): M
Score for Test A: 12
Score for Test B: 7
The average of the two test scores was 9.50
```

```
Gender (F for Female / M for Male): F
Score for Test A: 12
Score for Test B: 18
The average of the two test scores was 15.00
:
```

```
The average score for Test B (score > 15) for female student was xxx
The overall average score for Test A was xxx
The overall average score for Test B was xxx
```

APPENDIX G

Average rainfall

You have been asked to write a simple program to help a clerk to work with data on daily rainfall figures for several weeks. The program should calculate and display the average rainfall just for those days that it rained each week. This example illustrates a program that will read in total rainfall figures (in mm) for each day over the past few weeks. The program should work in situations such that the clerk should be able to decide number of weeks the program should calculate the average. Assume that the user will enter a valid input. A sample run is as follows:

```
Rainfall report for: 2 week(s)

Total rainfalls (in mm) for week 1 day 1 => 23.5

Total rainfalls (in mm) for week 1 day 2 => 15

Total rainfalls (in mm) for week 1 day 3 => 0
:

The average rainfall per rainy day for week 1 was: 16.0
```

Prompts and description of each plan

Loop entry condition, counter-controlled plan

Explain the condition under which the loop stops. The loop stops when the condition `day <= 7` eventually becomes false after the body of the loop has been executed 7 times and the final value of counter is 8.

Hint 1: Please refer to counter loop plan.
Hint 2: Condition under which the loop terminates.

Running total loop plan

Explain what happens if the user entered 5 for the total rainfalls of the day. The condition `(rainfalls > 0)` evaluates to true. The loop updates the value of `total_rainfalls` by adding 5 to it and increments the value of `total_rainydays` by 1.

Hint 1: Updates that are to take place when certain condition is met.
Hint 2:

Counter loop plan

Explain the role of counter loop variable `day_counter`. The variable represents the day number (i.e. 1 represents Monday and so on). The statement `day_counter++` updates the value of `day_counter` from 1 to 7 and increases its value by one each time through the loop.

Hint 1: The role of counter loop variable, `day_counter`.
Hint 2: Keep track the number of days in a week, from 1 to 7.

APPENDIX G

Guard and division plan

Explain the role of guard and division plan. The role of this plan is to check whether or not `total_rainydays` is 0 before dividing `total_rainfalls` by `total_rainydays`. Dividing by 0 causing run time error and will immediately stop the program. Therefore, if `total_rainydays` is 0, the program prints “No average computed”. Otherwise, an appropriate message with the value of average will be printed.

Hint 1: Division by zero.

Modification / reflection exercise

Modify the above program so that the program additionally calculates and displays at its end the average rainfall for those days that it rained in the whole period.

```
Rainfall report for: 3 week(s)

Total rainfalls (in mm) for week 1 day 1 => 13.5

Total rainfalls (in mm) for week 1 day 2 => 15.5

Total rainfalls (in mm) for week 1 day 3 => 23
:

The average rainfall per rainy day for week 1 was: xxx

The overall average rainfall per rainy day was: xxx
```

Average rainfall problem was adapted from Java Gently: Programming Principles Explains by Judy Bishop, 2nd edition, Addison-Wesley (1998).

APPENDIX G

Photocopy machine

You are hired by the Danial Bistari Bookshop to develop a program that accepts money for the company's self-service photocopier. The cost per copy for an A4 black and white document is 10 cents and for a colour document is 20 cents. The program should prompt the user to type in a letter B or C, representing black and white documents and colour documents respectively. Note that, the user must enter N to stop making further photocopying choices. The program should then compute the cost of the photocopying given the types of document and the number of copies the user has chosen, and calculate and give back the total change (if any). The photocopier accepts 50 cent, 20 cent, and 10 cent coin only. The machine can be used when at least the total cost of the photocopying is deposited. Thus the machine must keep track of the amount of money deposited by the user one coin at a time.

Please make a selection by choosing a letter for document type:

B. Black and white 10 cents
C. Colour 20 cents

This machine accepts 50 cent, 20 cent, and 10 cent coin only.

Document type? (N to stop): B

Number of copies: 2

Total amount due: 0.20

Document type? (N to stop): C

Number of copies: 2

Total amount due: 0.60

Document type? (N to stop): N

Please insert coins: 10

Amount of money deposited: 0.10

Please insert coins: 5

This machine accepts 50 cent, 20 cent, and 10 cent coin only.

Amount of money deposited: 0.10

Please insert coins: 20

Amount of money deposited: 0.30

Please insert coins: 50

Amount of money deposited: 0.80

Your change is: 0.20

Thank you

Prompts and description of each plan

Loop entry condition, sentinel-controlled

Explain what is meant by the expression `(option != SENTINEL)` in the while statement. The expression ensures that the body of the loop is repeated as long as the option entered by the user does not equal to SENTINEL value (i.e. N). That is, a check is made to see if further photocopying option is required before getting total money from the user.

Hint 1: The test to see if further photocopying option is required.

APPENDIX G

Running total and limit plan

The value of `payment` at which the loop starts is 0. Explain how the value of `payment` is updated through the loop. The amount of `payment` is increased by 0.50 cent, 0.20 cent, or 0.10 cent each time through the loop, depending on the user input. The loop terminates when `payment` exceeds `total_amount_due`.

Hint 1: Condition under which the value of `payment` is updated.

Valid data entry plan

Explain why the valid data entry plan is needed. The plan is needed to ensure that the user enter a valid coin. If an invalid coin is entered, an appropriate message is displayed and the `switch` statement terminates. The `while (total_amount_due > payment)` loop structure executes again and the user is prompted to enter a valid coin.

Hint 1: Please refer to running total and limit plan.

Hint 2: Check to see if a coin entered is valid.

Modification / reflection exercise

Modify the above program so that the user can deposit more money into the vending machine than needed. When the user wishes to discontinue, '-1' must be entered, however if the amount of money deposited to this point is not enough, a message "Please deposit more money" will be displayed.

Please make a selection by choosing a letter for document type:

B. Black and white 10 cents
C. Colour 20 cents

This machine accepts 50 cent, 20 cent, and 10 cent coin only.

Document type? (N to stop): C

Number of copies: 3

Total amount due: 0.60

Document type? (N to stop): N

Please insert coins: 20

Amount of money deposited: 0.20

Please insert coins: -1

Please insert more money....

Amount of money deposited: 0.20

Please insert coins: 10

Amount of money deposited: 0.30

Please insert coins: 5

This machine accepts 50 cent, 20 cent, and 10 cent coin only.

Amount of money deposited: 0.30

Please insert coins: 50

Amount of money deposited: 0.80

Please insert coins: -1

Your change is: 0.20

Thank you

APPENDIX G

Coffee machine

Kopi Umairah is a company that sells a range of coffee products via a vending machine. The machine has three choices of coffee, whose prices are as follows:

Latte	RM1.30
Cappuccino	RM1.50
Espresso	RM1.90

This example illustrates a vending machine program that prompts the user to type in L, C, or E representing Latte, Cappuccino, or Espresso, respectively. The machine accepts 50 cent, 20 cent, and 10 cent coins only. Coffee will be dispensed when at least the cost of the chosen coffee has been deposited. Thus the machine must keep track of the amount of money deposited by the user one coin at a time, calculate and give back the total change (if any). Finally, the machine should prompt the user to make another selection after a coffee has been purchased. Note that only one choice of coffee can be purchased within each transaction. Assume that the user will enter a valid input.

The sample run is as follow:

```
Please make a selection by choosing a letter for coffee:
-----
L. Latte          RM1.30
C. Cappuchino    RM1.50
E. Espresso      RM1.90

This machine accepts 50 cent, 20 cent, and 10 cent coin only.

Your selection of coffee? (N to stop the program): L
Amount due: 1.30
Please insert coin: 50
Amount of money deposited: 0.50
Please insert coin: 50
Amount of money deposited: 1.00
Please insert coin: 50
Amount of money deposited: 1.50
Your coffee is ready. Balance: 0.20

Another selection of coffee? (N to stop the program): C
Amount due: 1.50
Please insert coin: 10
Amount of money deposited: 0.10
Please insert coin: 20
Amount of money deposited: 0.30
Please insert coin: 50
Amount of money deposited: 0.80
Please insert coin: 50
Amount of money deposited: 1.30
Please insert coin: 50
Amount of money deposited: 1.80
Your coffee is ready. Balance: 0.30

Another selection of coffee? (N to stop the program): N
```

APPENDIX G

Prompts and description of each plan

Loop entry condition, sentinel-controlled

Explain the reasoning behind the expression `while (coffee != 'N')`. The expression is checked by comparing the `coffee` selection entered by the user with the sentinel value (i.e. N). If the user entered L, C or E, the body of the loop executes. Once a coffee has been purchased, the program prompts the user to make another selection of coffee. The expression `coffee != 'N'` is check again and if N is entered, the loop terminates.

Hint 1: Actions that are to take place after the expression has been evaluated.

Running total and limit plan

Explain the relationship between the `amount_due` and the `payment`. The values of the variables are used to evaluate the expression `(amount_due > payment)` in the while statement. The loop keeps track of the amount of `payment` received from the user and the loop continues to repeat until `payment` exceeds `amount_due`.

Hint 1: Keeps track of the amount of money deposited.

Hint 2: The loop continues until at least cost of photocopying has been deposited.

Valid data entry plan

Assume that the user wants to buy Latte and the amount of `payment` deposited so far is RM1.10. What would happen next, if the user entered 5 cent. An appropriate message is displayed and the `switch` statement terminates. The `while (amount_due > payment)` loop executes again, the user is prompted to insert a valid coin and the current value of coin is added to `payment`. When at least the cost of Latte has been deposited, the loop terminates.

Hint 1: Actions that are to take place when invalid coin is entered.

Modification / reflection exercise

Modify the above program so that the user is allowed to purchase another selection of coffee with the total change from the previous transaction. The user must deposit the remaining cost of the new coffee into the machine before receiving a coffee.

Please make a selection by choosing a letter for coffee:

L. Latte RM1.30
C. Cappuchino RM1.50
E. Espresso RM1.90

This machine accepts 50 cent, 20 cent, and 10 cent coin only.

Your selection of coffee? (N to stop the program): C

Amount due: 1.50

Please insert coin: 50

Amount of money deposited: 0.50

Please insert coin: 50

Amount of money deposited: 1.00

Please insert coin: 10

APPENDIX G

Amount of money deposited: 1.10
Please insert coin: 50
Amount of money deposited: 1.60
Your coffee is ready. Balance: 0.10

Another selection of coffee? (N to stop the program): L
Amount due: 1.30
Please insert coin: 50
Amount of money deposited: 0.60
Please insert coin: 50
Amount of money deposited: 1.10
Please insert coin: 10
Amount of money deposited: 1.20
Please insert coin: 10
Amount of money deposited: 1.30
Your coffee is ready. Balance: 0.00

Another selection of coffee? (N to stop the program): N

Vending machine problems were adapted from *An Introduction to Programming Using C++* by Kenneth C. Mansfield Jr. and James L. Antonakos, Prentice Hall (1997).

APPENDIX G

Coin tossing

Write an application that lets the user play a coin-tossing guessing game. The aim of the game is for the user to guess correctly the computer-simulated coin-toss. A game consists of a maximum of seven coin tosses. To win the game, the user must make five correct guesses in that seven, but not necessarily in a row. The user loses the game immediately if she makes three incorrect guesses altogether. The program starts by prompting the user to type in a number representing a coin-side. Assume that 1 represents Heads and 2 represents Tails. When the user makes a guess, the program replies with a message depending on whether the user has made a correct or incorrect guess. Finally, the program prints an appropriate message depending on whether the user won or lost the game. If the user won the game, the program displays a message indicating how many rounds the user took to win the game. Otherwise the program displays a message indicating how many correct guesses the user made prior to losing the game.

A sample run is as follows:

```
Your choice? (1. Heads or 2. Tails): 1
Chosen side of the coin: 1
Face-up side of the coin: 2
Incorrect guess.
```

```
Your choice? (1. Heads or 2. Tails): 2
Chosen side of the coin: 2
Face-up side of the coin: 1
Incorrect guess.
```

```
Your choice? (1. Heads or 2. Tails): 1
Chosen side of the coin: 1
Face-up side of the coin: 2
Incorrect guess.
```

You lost. You only made 0 correct guess(es) prior to losing the game.

```
Your choice? (1. Heads or 2. Tails): 1
Chosen side of the coin: 1
Face-up side of the coin: 2
Incorrect guess.
```

```
Your choice? (1. Heads or 2. Tails): 1
Chosen side of the coin: 1
Face-up side of the coin: 1
Correct guess.
```

```
Your choice? (1. Heads or 2. Tails): 2
Chosen side of the coin: 2
Face-up side of the coin: 1
Incorrect guess.
```

```
Your choice? (1. Heads or 2. Tails): 2
Chosen side of the coin: 2
Face-up side of the coin: 1
Incorrect guess.
```

You lost. You only made 1 correct guesses prior to losing the game.

```
Your choice? (1. Heads or 2. Tails): 1
```

APPENDIX G

Chosen side of the coin: 1
Face-up side of the coin: 2
Incorrect guess.

Your choice? (1. Heads or 2. Tails): 1
Chosen side of the coin: 1
Face-up side of the coin: 1
Correct guess.

Your choice? (1. Heads or 2. Tails): 2
Chosen side of the coin: 2
Face-up side of the coin: 2
Correct guess.

Your choice? (1. Heads or 2. Tails): 1
Chosen side of the coin: 1
Face-up side of the coin: 1
Correct guess.

Your choice? (1. Heads or 2. Tails): 2
Chosen side of the coin: 2
Face-up side of the coin: 2
Correct guess.

Your choice? (1. Heads or 2. Tails): 1
Chosen side of the coin: 1
Face-up side of the coin: 1
Correct guess.

You won the game. It took 6 rounds to win the game.

Prompts and description of each plan

Flag reset plan

The value of `flag` is initialized to `TRUE`. Explain the condition under which the `flag` remains true. The flag remains true when the user has reached maximum of seven coin tosses and eventually made less than five correct guesses. `flag` is reset to `FALSE` when `correct_guess_count` reaches five.

Hint 1: Please refer to loop exit condition, counter- and flag-controlled.

Hint 2: `flag` becomes false when the user has made five correct guesses within a maximum of seven coin tosses.

Loop exit condition, counter- and flag-controlled plan

The expression `while (round_count < MAX_ROUNDS && flag == TRUE)` is used here for two purposes. Explain. To check whether the user has reached maximum of seven coin tosses, `round_count < MAX_ROUNDS` or the user has made five correct guesses within seven coin tosses, `flag == TRUE`. If both conditions are true, the loop will continue. However, if either condition is false or both conditions are false, the loop will terminate.

APPENDIX G

- Hint 1: The do...while loop count number of correct guesses the user made within a maximum of seven coin tosses.
- Hint 2: The loop terminates if either condition or both conditions are false.

Counter loop plan

Explain the condition under which the loop is to continue with regard to the value of `round_count`. The loop continues as long as the value of `round_count` is less than `MAX_ROUNDS`. `round_count` increases its value by one each time through the loop. The loop stops when the value of `round_count` reaches seven.

- Hint 1: A game consists of at most seven coin tosses.
- Hint 2: `round_count` ensures that each repetition makes progress towards loop termination.

Running total loop plan

Explain what happens if the value of `coin_side` and `face_up_side` is 2. The expression in the do...while statement, `if(coin_side == face_up_side)` evaluates to true and the loop will increment the value of `correct_guess_count` by 1.

- Hint 1: Keeps track number of correct guesses the user has made.
- Hint 2: Update that is to take place when the expression is true.

Modification / reflection exercise

Modify the above program so that the user must make at least two correct guesses in a row in order to win the game. There is now no maximum of seven rounds, but note that the game terminates the first time that the user makes an incorrect guess. At that point the program should display an appropriate message depending on whether the user has won or lost. If the user won the game, there should be a message indicating the number of correct guesses in a row. Otherwise, the program displays a message “You lost”.

Your choice? (1. Heads or 2. Tails): 2
Chosen side of the coin: 2
Face-up side of the coin: 2
Correct guess.

Your choice? (1. Heads or 2. Tails): 2
Chosen side of the coin: 2
Face-up side of the coin: 2
Correct guess.

Your choice? (1. Heads or 2. Tails): 2
Chosen side of the coin: 2
Face-up side of the coin: 1
Incorrect guess.

You won the game. 2 correct guesses in a row.

Your choice? (1. Heads or 2. Tails): 1
Chosen side of the coin: 1
Face-up side of the coin: 1
Correct guess.

APPENDIX G

Your choice? (1. Heads or 2. Tails): 1
Chosen side of the coin: 1
Face-up side of the coin: 2
Incorrect guess.

You lost.

Your choice? (1. Heads or 2. Tails): 2
Chosen side of the coin: 2
Face-up side of the coin: 2
Correct guess.

Your choice? (1. Heads or 2. Tails): 2
Chosen side of the coin: 2
Face-up side of the coin: 2
Correct guess.

Your choice? (1. Heads or 2. Tails): 2
Chosen side of the coin: 2
Face-up side of the coin: 2
Correct guess.

Your choice? (1. Heads or 2. Tails): 2
Chosen side of the coin: 2
Face-up side of the coin: 1
Incorrect guess.

You won the game. 3 correct guesses in a row.

Coin-tossing game-based problem was adapted from C by Dissection: The Essentials of C Programming by Al Kelly and Ira Pohl, 3rd edition, Addison-Wesley (1996).

APPENDIX G

Rock, paper, scissors

Write a program that lets the user (Player B) play a hand-game called Rock-Paper-Scissors with the computer (Player A). The aim of the game is to choose a hand-sign signifying Rock, Paper, or Scissors to defeat the computer-generated hand-sign. The maximum number of tries allowed is five. The game terminates if the user wins a round before the five tries are used up. The program displays a simple list for the user to choose a number representing his/her hand-sign. That is, number 1, 2, and 3 represents Rock, Paper, and Scissors, respectively. The winner is determined as follows:

Rock wins against scissors because it blunts scissors.

Scissors wins against paper because it cuts paper.

Paper wins against rock because it covers rock.

The game is tied if both the user and the computer choose the same hand-sign. When the user chooses a hand-sign, the program replies with an appropriate message depending on whether the game is tied, or the user won or lost. Finally, there should be a message indicating the number of tries it took for the user to win the game or a message indicating the user has reached the maximum number of tries along with number of draw(s) and lost(s) the user made.

Your choice? (1. Rock, 2. Paper or 3. Scissors): 1

Player A: 1 Player B: 1

It's a draw.

Your choice? (1. Rock, 2. Paper or 3. Scissors): 2

Player A: 2 Player B: 2

It's a draw.

Your choice? (1. Rock, 2. Paper or 3. Scissors): 3

Player A: 3 Player B: 3

It's a draw.

Your choice? (1. Rock, 2. Paper or 3. Scissors): 1

Player A: 2 Player B: 1

You lost.

Your choice? (1. Rock, 2. Paper or 3. Scissors): 2

Player A: 1 Player B: 2

You won.

You took 5 tries to win the game.

Your choice? (1. Rock, 2. Paper or 3. Scissors): 2

Player A: 2 Player B: 2

It's a draw.

Your choice? (1. Rock, 2. Paper or 3. Scissors): 3

Player A: 1 Player B: 3

APPENDIX G

You lost.

Your choice? (1. Rock, 2. Paper or 3. Scissors): 1

Player A: 2 Player B: 1

You lost.

Your choice? (1. Rock, 2. Paper or 3. Scissors): 2

Player A: 2 Player B: 2

It's a draw.

Your choice? (1. Rock, 2. Paper or 3. Scissors): 3

Player A: 1 Player B: 3

You lost.

You have reached maximum number of tries, ended with 2 draw(s) and 3 lost(s).

Prompts and description of each plan

Flag reset plan

Explain what happens if the user won the game when the current value of `try_count` is 3. The statement `flag = FALSE` reset the value of `flag` to `FALSE` thus making the expression `while (try_count < MAX_TRIES_ALLOWED && flag == TRUE)` becoming false and eventually stops the loop.

Hint 1: Please refer to loop exit condition, counter- and flag-controlled.

Hint 2: `flag` becomes false when the user has made correct guess within a maximum number of tries.

Loop exit condition, counter- and flag-controlled plan

Explain the reasoning behind the expression `while (try_count < MAX_TRIES_ALLOWED && flag == TRUE)`. The expression ensures that the body of the loop is repeated as long as the number of tries is not spent and the user is not yet won the game. The loop stops when `try_count` reaches five or the user has won the game or when both conditions evaluate to false.

Hint 1: The do...while loop keeps track number of tries the user has spent.

Hint 2: The loop terminates if either or both conditions are false.

Counter loop plan

Explain why is the statement `try_count = try_count+1` needed within the while loop. The statement ensures that each iteration makes progress toward the condition `try_count < MAX_TRIES_ALLOWED` becoming false. That is, the statement keeps track of the number of tries the user has spent.

Hint 1: Maximum number of tries allowed is five.

Hint 2: `try_count` ensures that each repetition makes progress towards loop termination.

APPENDIX G

Running total loop plan

Explain how the loop keeps track number of draw(s) or lost(s) the user made. The loops keep track number of draw(s) or lost(s) the user made by checking whether certain conditions are met, depending on the user's choice and the computer's choice. For example, if both the user and the computer choose Rock, the loop will update the value of `draw_count` by 1. However, if the computer chooses Paper and the user chooses scissors, the loop will update the value of `lost_count` by 1.

Hint 1: Condition under which the values of running total loop variable is updated with the current value.

Modification / reflection exercise

Instead of allowing the player to play up to maximum number of tries, modify the program so that the player is prompted (at the end of every game) to see if they want to have another go or not. Finally there should be a message indicating the number of times the user won out of number of games played along with number of draw(s) and lost(s) the user made.

```
Your choice? (1. Rock, 2. Paper or 3. Scissors): 1
Player A: 2           Player B: 1
```

```
You lost.
```

```
Do you want to play another game or not? (Y/N): Y
```

```
Your choice? (1. Rock, 2. Paper or 3. Scissors): 2
Player A: 1           Player B: 2
```

```
You won.
```

```
Do you want to play another game or not? (Y/N): Y
```

```
Your choice? (1. Rock, 2. Paper or 3. Scissors): 3
Player A: 3           Player B: 3
```

```
It's a draw.
```

```
Do you want to play another game or not? (Y/N): N
```

```
Number of times that you won was 1 out of 3 game(s) played and ended with 1 draw(s) and 1 lost(s)
```

Rock, paper, scissors game-based problem was adapted from Java Gently: Programming Principles Explains by Judy Bishop, 2nd edition, Addison-Wesley (1998).

APPENDIX G

Example solution

Average score

```
import java.util.Scanner;

public class AverageScore {

    public static void main(String args[]){

        int counter, scoreA, scoreB, female_count;
        double total_score, female_scoreB;
        char gender;

        counter = 0;
        total_score = 0.0;
        female_count = 0;
        female_scoreB = 0.0; // Initialisation plan

        Scanner keyboard = new Scanner(System.in);

        while(counter < 20 ){ // Loop entry condition, counter-controlled plan

            System.out.print("Gender (F for Female / M for Male): ");
            gender = keyboard.next().charAt(0);

            System.out.print("Score for Test A: ");
            scoreA = keyboard.nextInt();

            System.out.print("Score for Test B: ");
            scoreB = keyboard.nextInt();

            total_score = scoreA + scoreB;

            System.out.printf("The average of the two test scores was %.2f\n" , total_score/2);

            if (gender == 'F' && scoreB > 15){
                female_count++;
                female_scoreB = female_scoreB + scoreB; // Running total loop plan
            }

            counter++; // Counter loop plan
        }

        if (female_count != 0) // Guard and division plan
            System.out.printf("\nThe average score for Test B (score > 15) for female student was %.2f\n" ,
                female_scoreB/female_count);
        else
            System.out.println("\nNo average calculated\n");
        }
    }
```

===== End of average score solution =====

APPENDIX G

Average score (2)

```
import java.util.Scanner;

public class AverageScore2 {

    public static void main(String args[]){

        int counter, scoreA, scoreB, female_count;
        double total_score, female_scoreB, total_scoreA, total_scoreB;
        char gender;

        counter = 0;
        total_score = 0.0;
        female_count = 0;
        female_scoreB = 0.0; // Initialisation plan
        total_scoreA = 0.0;
        total_scoreB = 0.0;

        Scanner keyboard = new Scanner(System.in);

        while(counter < 20){ // Loop entry condition, counter-controlled plan

            System.out.print("Gender (F for Female / M for Male): ");
            gender = keyboard.next().charAt(0);

            System.out.print("Score for Test A: ");
            scoreA = keyboard.nextInt();

            System.out.print("Score for Test B: ");
            scoreB = keyboard.nextInt();

            total_scoreA = total_scoreA + scoreA;
            total_scoreB = total_scoreB + scoreB;
            total_score = scoreA + scoreB;

            System.out.printf("The average of the two test scores was %.2f\n" , total_score/2);

            if (gender == 'F' && scoreB > 15){
                female_count++;
                female_scoreB = female_scoreB + scoreB; // Running total loop plan
            }

            counter++; // Counter loop plan
        }

        if (female_count != 0) // Guard and division plan
            System.out.printf("\nThe average score for Test B (score > 15) for female student was %.2f\n" ,
                female_scoreB/female_count);
        else
            System.out.println("\nNo average calculated\n");

        System.out.printf("The overall average score for Test A was %.2f\n" , total_scoreA/20);

        System.out.printf("The overall average score for Test B was %.2f\n" , total_scoreB/20);
    }
}
```

===== End of average score (2) solution =====

APPENDIX G

Average rainfall

```
import java.util.Scanner;

public class AverageRainfall{

    public static void main(String args[]){

        int week, week_counter, day_counter, total_rainydays;
        double rainfall, total_rainfalls;

        week_counter = 1; // Initialisation plan

        Scanner keyboard = new Scanner(System.in);

        System.out.print("Rainfall report for: ");
        week = keyboard.nextInt();
        System.out.println("week(s)");

        while (week_counter <= week){ // Loop entry condition, counter-controlled plan

            day_counter = 1;
            total_rainydays = 0;
            total_rainfalls = 0.0; // Initialisation plan

            while (day_counter <= 7){ // Loop entry condition, counter-controlled plan

                System.out.print("\nTotal rainfalls (in mm) for week " + week_counter + " day " + day_counter + " =>");
                rainfall = keyboard.nextDouble();

                if (rainfall > 0){
                    total_rainfalls = total_rainfalls + rainfall; // Running total loop plan
                    total_rainydays = total_rainydays + 1;
                }

                day_counter++; // Counter loop plan
            }

            if (total_rainydays != 0) // Guard and division plan
                System.out.printf("\nThe average rainfall per rainy day for week " + week_counter + " was:
                %.2f\n", total_rainfalls/total_rainydays);
            else
                System.out.println("\nNo average calculated.\n");

            week_counter++; // Counter loop plan
        }
    }

    == == == End of average rainfall solution == == ==
```

APPENDIX G

Average rainfall (2)

```
import java.util.Scanner;

public class AverageRainfall2{

    public static void main(String args[]){

        int week, week_counter, day_counter, total_rainydays, overall_rainydays;
        double rainfall, total_rainfalls, overall_rainfalls;

        week_counter = 1;
        overall_rainydays = 0;
        overall_rainfalls = 0.0; // Initialisation plan

        Scanner keyboard = new Scanner(System.in);

        System.out.print("Rainfall report for: ");
        week = keyboard.nextInt();
        System.out.println("week(s)");

        while (week_counter <= week){ // Loop entry condition, counter-controlled plan

            day_counter = 1;
            total_rainydays = 0;
            total_rainfalls = 0.0; // Initialisation plan

            while (day_counter <= 7){ // Loop entry condition, counter-controlled plan

                System.out.print("\nTotal rainfalls (in mm) for week " + week_counter + " day " + day_counter + " =>");
                rainfall = keyboard.nextDouble();

                if (rainfall > 0){
                    total_rainfalls = total_rainfalls + rainfall; // Running total loop plan
                    total_rainydays = total_rainydays + 1;
                    overall_rainfalls = overall_rainfalls + rainfall;
                    overall_rainydays = overall_rainydays + 1;
                }

                day_counter++; // Counter loop plan
            }

            if (total_rainydays != 0) // Guard and division plan
                System.out.printf("\nThe average rainfall per rainy day for week " + week_counter + " was:
                %.2f\n", total_rainfalls/total_rainydays);
            else
                System.out.println("\nNo average calculated.\n");

            week_counter++; // Counter loop plan
        }

        if (overall_rainydays != 0) // Guard and division plan
            System.out.printf("\nThe overall average rainfall per rainy day was: %.2f",
            overall_rainfalls/overall_rainydays);
        else
            System.out.println("\nNo overall average calculated.");
    }
}
```

==== End of average rainfall (2) solution ====

APPENDIX G

Photocopy machine

```
import java.util.Scanner;

public class PhotocopyMachine{

    public static void main(String args[]){

        final char SENTINEL = 'N';
        int copies, coins;
        float amount_due, total_amount_due, payment;
        char option;

        total_amount_due = 0;
        payment = 0; // Initialisation plan

        Scanner keyboard = new Scanner(System.in);

        System.out.println("Please make a selection by choosing a letter for document type:");
        System.out.println("-----");
        System.out.println("B. Black and white 10 cents");
        System.out.println("C. Colour      20 cents\n");
        System.out.println("This machine accepts 50 cent, 20 cent, and 10 cent coin only.\n");

        System.out.print("Document type? (N to stop): ");
        option = keyboard.next().charAt(0);

        while (option != SENTINEL){ // Loop entry condition, sentinel-controlled plan

            amount_due = 0; // Initialisation plan

            System.out.print("Number of copies: ");
            copies = keyboard.nextInt();

            if (option == 'B')
                amount_due = (float)(0.10 * copies);
            else
                amount_due = (float)(0.20 * copies);

            total_amount_due = total_amount_due + amount_due;

            System.out.printf("Total amount due: %.2f\n", total_amount_due);

            System.out.print("Document type? (N to stop): ");
            option = keyboard.next().charAt(0);
        }

        while (payment < total_amount_due){ // Running total and limit plan

            System.out.print("Please insert coins: ");
            coins = keyboard.nextInt();

            switch(coins){
                case 50 : payment = (float)(payment + 0.50);
                    break;
                case 20 : payment = (float)(payment + 0.20);
                    break;
            }
        }
    }
}
```

APPENDIX G

```
        case 10 : payment = (float)(payment + 0.10);
        break;
        default : System.out.println("This machine accepts 50 cent, 20 cent, and 10 cent coin only.");
        break; // Valid data entry plan
    }

    System.out.printf("Amount of money deposited: %.2f\n", payment);
}

if(payment > total_amount_due){
    System.out.printf("\nYour change is: %.2f\n", payment - total_amount_due);
    System.out.println("Thank you");
}
else
    System.out.println("Thank you");
}
}
```

===== End of photocopy machine solution =====

APPENDIX G

Photocopy machine (2)

```
import java.util.Scanner;

public class PhotocopyMachine4{

    public static void main(String args[]){

        final char SENTINEL = 'N';
        int copies, coins;
        float amount_due, total_amount_due, payment;
        char option;

        total_amount_due = 0;
        payment = 0; // Initialisation plan

        Scanner keyboard = new Scanner(System.in);

        System.out.println("Please make a selection by choosing a letter for document type:");
        System.out.println("-----");
        System.out.println("B. Black and white 10 cents");
        System.out.println("C. Colour      20 cents\n");
        System.out.println("This machine accepts 50 cent, 20 cent, and 10 cent coin only.\n");

        System.out.print("Document type? (N to stop): ");
        option = keyboard.next().charAt(0);

        while (option != SENTINEL){ // Loop entry condition, sentinel-controlled plan

            amount_due = 0; // Initialisation plan

            System.out.print("Number of copies: ");
            copies = keyboard.nextInt();

            if (option == 'B')
                amount_due = (float)(0.10 * copies);
            else
                amount_due = (float)(0.20 * copies);

            total_amount_due = total_amount_due + amount_due;

            System.out.printf("Total amount due: %.2f\n", total_amount_due);

            System.out.print("Document type? (N to stop): ");
            option = keyboard.next().charAt(0);
        }

        if (total_amount_due > payment){

            System.out.print("Please insert coins: ");
            coins = keyboard.nextInt();

            while ((coins != -1) || (payment < total_amount_due)) { // Running total and limit plan

                switch(coins){
                    case 50 : payment = (float)(payment + 0.50);
                        break;
                    case 20 : payment = (float)(payment + 0.20);
                        break;
                }
            }
        }
    }
}
```

APPENDIX G

```
        case 10 : payment = (float)(payment + 0.10);
        break;
        case -1 : System.out.println("Please insert more money....");
        break;
        default : System.out.println("This machine accepts 50 cent, 20 cent, and 10 cent coin only.");
        break; // Valid data entry plan
    }

    System.out.printf("Amount of money deposited: %.2f\n", payment);

    System.out.print("Please insert coins: ");
    coins = keyboard.nextInt();
    }
}

if(payment > total_amount_due){
    System.out.printf("\nYour change is: %.2f\n", payment - total_amount_due);
    System.out.println("Thank you");
}
else
    System.out.println("Thank you");
}
}
```

===== End of photocopy machine (2) solution =====

APPENDIX G

Coffee machine

```
import java.util.Scanner;

public class CoffeeMachine{

    public static void main(String args[]){

        final double LATTE = 1.30, CAPPUCHINO = 1.50, ESPRESSO = 1.90;
        double amount_due, payment;
        int coins;
        char coffee;

        Scanner keyboard = new Scanner(System.in);

        System.out.println("\nPlease make a selection by choosing a letter for coffee:");
        System.out.println("-----");
        System.out.println("L. Latte    RM1.30");
        System.out.println("C. Cappuchino  RM1.50");
        System.out.println("E. Espresso  RM1.90\n");
        System.out.println("This machine accepts 50 cent, 20 cent, and 10 cent coin only.\n");

        System.out.print("Your selection of coffee? (N to stop the program): ");
        coffee = keyboard.next().charAt(0);

        while (coffee != 'N'){ // Loop entry condition, sentinel-controlled plan

            amount_due = 0.0;
            payment = 0.0; // Initialisation plan

            if (coffee == 'L')
                amount_due = LATTE;
            else if (coffee == 'C')
                amount_due = CAPPUCHINO;
            else if (coffee == 'E')
                amount_due = ESPRESSO;

            System.out.printf("Amount due: %.2f\n", amount_due);

            while (payment < amount_due){ // Running total and limit plan

                System.out.print("Please insert coin: ");
                coins = keyboard.nextInt();

                switch(coins){
                    case 50 : payment += 0.50;
                        break;
                    case 20 : payment += 0.20;
                        break;
                    case 10 : payment += 0.10;
                        break;
                    default : System.out.println("This machine accepts 50 cent, 20 cent, and 10 cent coin only.");
                        break; // Valid data entry plan
                }

                System.out.printf("Amount of money deposited: %.2f\n", payment);
            }
        }
    }
}
```

APPENDIX G

```
System.out.printf("\nYour coffee is ready. Balance: %.2f\n", payment - amount_due);

System.out.print("\nAnother selection of coffee? (N to stop the program): ");
coffee = keyboard.next().charAt(0);
}
}
}
```

===== End of coffee machine solution =====

APPENDIX G

Coffee machine (2)

```
import java.util.Scanner;

public class CoffeeMachine2{

    public static void main(String args[]){

        final double LATTE = 1.30, CAPPUCHINO = 1.50, ESPRESSO = 1.90;
        double amount_due, payment, balance;
        int coins;
        char coffee;

        balance = 0.0; // Initialisation plan

        Scanner keyboard = new Scanner(System.in);

        System.out.println("\nPlease make a selection by choosing a letter for coffee:");
        System.out.println("-----");
        System.out.println("L. Latte    RM1.30");
        System.out.println("C. Cappuchino RM1.50");
        System.out.println("E. Espresso  RM1.90\n");
        System.out.println("This machine accepts 50 cent, 20 cent, and 10 cent coin only.\n");

        System.out.print("Your selection of coffee? (N to stop the program): ");
        coffee = keyboard.next().charAt(0);

        while (coffee != 'N'){ // Loop entry condition, sentinel-controlled plan

            amount_due = 0.0;
            payment = 0.0; // Initialisation plan
            payment = balance;

            if (coffee == 'L')
                amount_due = LATTE;
            else if (coffee == 'C')
                amount_due = CAPPUCHINO;
            else if (coffee == 'E')
                amount_due = ESPRESSO;

            System.out.printf("Amount due: %.2f\n", amount_due);

            while (payment < amount_due){ // Running total and limit plan

                System.out.print("Please insert coin: ");
                coins = keyboard.nextInt();

                switch(coins){
                    case 50 : payment += 0.50;
                        break;
                    case 20 : payment += 0.20;
                        break;
                    case 10 : payment += 0.10;
                        break;
                    default : System.out.println("This machine accepts 50 cent, 20 cent, and 10 cent coin only.");
                        break; // Valid data entry plan
                }

                System.out.printf("Amount of money deposited: %.2f\n", payment);
            }
        }
    }
}
```

APPENDIX G

```
balance = payment - amount_due;

System.out.printf("\nYour coffee is ready. Balance: %.2f\n", balance);

System.out.print("\nAnother selection of coffee? (N to stop the program): ");
coffee = keyboard.next().charAt(0);

if (coffee == 'N' && balance != 0.00)
    System.out.printf("Here is your balance: %.2f\n", balance);
}
}
}
```

===== End of coffee machine (2) solution =====

APPENDIX G

Coin tossing

```
import java.util.Scanner;

public class CoinTossing {

    public static void main(String args[]){

        final boolean TRUE = true, FALSE = false;
        final int MAX_ROUNDS = 7;
        int coin_side, face_up_side, correct_guess_count, incorrect_guess_count, round_count;
        boolean flag;
        double Y;

        flag = TRUE; // Initialisation plan
        correct_guess_count = 0;
        incorrect_guess_count = 0;
        round_count = 0;

        Scanner keyboard = new Scanner(System.in);

        do {

            System.out.print("Your choice? (1. Heads or 2. Tails):");
            coin_side = keyboard.nextInt();

            Y = Math.random();
            face_up_side = (int) Math.floor(Y*2)+1;

            System.out.println("Chosen side of the coin: " + coin_side);
            System.out.println("Face-up side of the coin: " + face_up_side);

            if (coin_side == face_up_side){
                System.out.println("Correct guess.\n");
                correct_guess_count = correct_guess_count + 1; // Running total loop plan
            }
            else
            {
                System.out.println("Incorrect guess.\n");
                incorrect_guess_count = incorrect_guess_count + 1; // Running total loop plan
            }

            round_count++; // Counter loop plan

            if ((correct_guess_count == 5)|| (incorrect_guess_count == 3))
                flag = FALSE; // Flag reset plan

        } while (round_count < MAX_ROUNDS && flag == TRUE);
        // Loop exit condition, counter- and flag-controlled plan

        if (correct_guess_count == 5)
            System.out.println("You won the game." + " It took " + round_count + " rounds to win the game.");
        else
            System.out.println("You lost." + " You only made " + correct_guess_count + " correct guess(es) prior to losing the game.");
    }
}
```

===== End of coin tossing solution =====

APPENDIX G

Coin tossing (2)

```
import java.util.Scanner;

public class CoinTossing2 {

    public static void main(String args[]){

        final boolean TRUE = true, FALSE = false;
        int coin_side, face_up_side, correct_guess_count;
        boolean flag;
        double Y;

        flag = TRUE; // Initialisation plan
        correct_guess_count = 0;

        Scanner keyboard = new Scanner(System.in);

        do {

            System.out.print("Your choice? (1. Heads or 2. Tails):");
            coin_side = keyboard.nextInt();

            Y = Math.random();
            face_up_side = (int) Math.floor(Y*2)+1;

            System.out.println("Chosen side of the coin: " + coin_side);
            System.out.println("Face-up side of the coin: " + face_up_side);

            if (coin_side == face_up_side){
                System.out.println("Correct guess.\n");
                correct_guess_count = correct_guess_count + 1; // Running total loop plan
            }
            else
                System.out.println("Incorrect guess.\n");

            if (coin_side != face_up_side)
                flag = FALSE; // Flag reset plan

        } while (flag == TRUE); // Loop exit condition, flag-controlled plan

        if (correct_guess_count >= 2 )
            System.out.println("You won the game. " + correct_guess_count + " correct guesses in a row.");
        else
            System.out.println("You lost.");
    }
}
```

===== End of coin tossing (2) solution =====

APPENDIX G

Rock, paper, scissors

```
import java.util.Scanner;

public class RockPaperScissors{
    public static void main(String args[]){

        final boolean TRUE = true, FALSE = false;
        final int MAX_TRIES_ALLOWED = 5;
        int computer_choice, user_choice, try_count, draw_count, lost_count;
        boolean flag;
        double Y;

        flag = TRUE; // Initialisation plan
        try_count = 0;
        draw_count = 0;
        lost_count = 0;

        Scanner keyboard = new Scanner(System.in);

        do {

            Y = Math.random();
            computer_choice = (int) Math.floor(Y*3)+1;

            try_count = try_count + 1; // Counter loop plan

            System.out.print("\nYour choice? (1. Rock, 2. Paper or 3. Scissors):");
            user_choice = keyboard.nextInt();

            if (computer_choice == user_choice){
                System.out.println("Player A: " + computer_choice + "\t Player B: " + user_choice);
                System.out.println("\nIt's a draw.");
                draw_count = draw_count + 1; // Running total loop plan
            }
            else
            if ((computer_choice==1 && user_choice==3)|| (computer_choice==2 &&
                user_choice==1)|| (computer_choice==3 && user_choice==2)){
                System.out.println("Player A: " + computer_choice + "\t Player B: " + user_choice);
                System.out.println("\nYou lost.");
                lost_count = lost_count + 1; // Running total loop plan
            }
            else
            if ((computer_choice==1 && user_choice==2)|| (computer_choice==2 &&
                user_choice==3)|| (computer_choice==3 && user_choice==1)){
                System.out.println("Player A: " + computer_choice + "\t Player B: " + user_choice);
                System.out.println("\nYou won.");
                flag = FALSE; // Flag reset plan
            }
        } while (try_count < MAX_TRIES_ALLOWED && flag == TRUE);
        // Loop exit condition, counter- and flag-controlled plan

        if (flag == TRUE)
            System.out.println("\nYou have reached maximum number of tries, ended with " + draw_count + "
                draw(s) and " + lost_count + " lost(s).");
        else
            System.out.println("\nYou took " + try_count + " tries to win the game.");
        }
    }
} // == End of coin rock, paper, scissors solution ==
```

APPENDIX G

Rock, paper, scissors (2)

```
import java.util.Scanner;

public class RockPaperScissors2{

    public static void main(String args[]){

        final boolean TRUE = true, FALSE = false;
        int computer_choice, user_choice, try_count, draw_count, lost_count, win_count;
        boolean flag;
        double Y;
        char answer;

        flag = TRUE; // Initialisation plan
        try_count = 0;
        draw_count = 0;
        lost_count = 0;
        win_count = 0;

        Scanner keyboard = new Scanner(System.in);

        do {

            Y = Math.random();
            computer_choice = (int) Math.floor(Y*3)+1;

            try_count = try_count + 1; // Counter loop plan

            System.out.print("\nYour choice? (1. Rock, 2. Paper or 3. Scissors):");
            user_choice = keyboard.nextInt();

            if (computer_choice == user_choice){
                System.out.println("Player A: " + computer_choice + "\t Player B: " + user_choice);
                System.out.println("\nIt's a draw.");
                draw_count = draw_count + 1; // Running total loop plan
            }
            else
            if ((computer_choice==1 && user_choice==3)||((computer_choice==2 &&
                user_choice==1)||((computer_choice==3 && user_choice==2)){
                System.out.println("Player A: " + computer_choice + "\t Player B: " + user_choice);
                System.out.println("\nYou lost.");
                lost_count = lost_count + 1; // Running total loop plan
            }
            else
            if ((computer_choice==1 && user_choice==2)||((computer_choice==2 &&
                user_choice==3)||((computer_choice==3 && user_choice==1)){
                System.out.println("Player A: " + computer_choice + "\t Player B: " + user_choice);
                System.out.println("\nYou won.");
                win_count = win_count + 1; // Running total loop plan
            }

            System.out.print("\nDo you want to play another game or not? (Y/N): ");
            answer = keyboard.next().charAt(0);
```


APPENDIX G

```
        if (answer == 'N')
            flag = FALSE; // Flag reset plan

    } while (flag == TRUE); // Loop exit condition, flag-controlled plan

    System.out.println("\nNumber of times that you won was " + win_count + " out of " + try_count + "
game(s) played and ended with " + draw_count + " draw(s) and " + lost_count + " lost(s)");
}
}
```

===== End of coin, rock, paper, scissors (2) solution =====

APPENDIX H

Transfer problems for the transfer phase

Problem 1¹

You are hired by Senjana Bus Services to develop a program that accepts money for the company's self-service ticket machine. The bus operates within two zones, namely Zone A and Zone B. The prices are shown below:

	Single	Return
Zone A	RM3.00	RM5.00
Zone B	RM1.00	RM3.00

The program should prompt the user to type in the letter A or B or N, representing Zone A, Zone B or no more tickets respectively, and a number for the journey type. That is, 1 represents a Single journey and 2 represents a Return journey. The program should compute the price of all the bus tickets, given the Zones the user wishes to go to and the types of journey. This price is computed once the user types N as the Zone. The machine accepts RM1, RM5 and RM10 notes only. The ticket will be dispensed when at least the price of the bus ticket has been inserted. Therefore, the machine must keep track of the amount of money inserted by the user. The program should calculate and give back the total change (if any).. Assume that the user always enters valid input. The sample run is as follows:

Please make a selection by choosing a letter for zone and a number for a journey

	1.Single	2.Return
Zone A	RM3.00	RM5.00
Zone B	RM1.00	RM3.00

This machine accepts RM1, RM5 and RM10 only

Zone (N to stop): B

Journey: 2

Amount due: RM3.00

Zone (N to stop): A

Journey: 1

Amount due: RM6.00

Zone (N to stop): N

Please insert RM notes: 1

Amount received: RM1.00

Please insert RM notes: 2

This machine accepts RM10, RM5, and RM1 only.

Please insert RM notes: 5

Amount received: RM6.00

Thank you

The time at which you have completed Problem 1 _____

Please rate your perceived mental effort on solving this problem (circle).

1. Very low mental effort
2. Low mental effort
3. Moderate mental effort
4. High mental effort
5. Very high mental effort

¹ Problem adapted from An Introduction to Programming Using C++ by Kenneth C. Mansfield Jr. and James L. Antonakos, Prentice Hall (1997).

APPENDIX H

Problem 2²

Write an application that will play Hi-Lo games with the user. The objective of the game is for the user to guess the computer-generated secret-number in the least number of tries. The secret number is an integer between 1 and 100, inclusive. When the user makes a guess, the program replies with Hi or Lo depending on whether the guess is higher or lower than the secret number. The maximum number of tries allowed for each game is six.

Enter a secret number (from 1 to 100): 10
Your guess is high.
Enter a secret number (from 1 to 100): 5
Your guess is low.
Enter a secret number (from 1 to 100): 6
Congratulations
You took 3 guesse(s).

Enter a secret number (from 1 to 100): 50
Your guess is high.
Enter a secret number (from 1 to 100): 40
Your guess is high.
Enter a secret number (from 1 to 100): 30
Your guess is high.
Enter a secret number (from 1 to 100): 20
Your guess is high.
Enter a secret number (from 1 to 100): 10
Your guess is low.
Enter a secret number (from 1 to 100): 15
Your guess is high.
You lost. Secret number was 13

The time at which you have completed Problem 2 _____

Please rate your perceived mental effort on solving this problem (circle).

1. Very low mental effort
2. Low mental effort
3. Moderate mental effort
4. High mental effort
5. Very high mental effort

² Problem taken from An introduction to object-oriented programming with Java by Wu, C. Thomas, 2nd Edition, McGraw-Hill Higher Education (2006).

APPENDIX H

Problem 3³

At the start of every month Emil deposits money into a saving account with an interest rate of 5% per annum. If he puts RM200 in at the start of the first month, by the end of that month, his savings amount becomes: $200 * (1 + 0.00417) = 200.83$. If he deposits a further RM100 at the start of the second month, by the end of the second month, his savings amount becomes: $(100 + 200.83) * (1 + 0.00417) = 302.09$ and so on. Note that, the interest rate is 0.00417 per month (that is, $0.05 / 12$).

Write a program that prompts the user at the start of each month to enter an amount to be deposited that month. This should continue until the user types -1 for the amount. The program should calculate and display the total amount in the account at the end of that month. The program should also display at its end the total interest earned since the start. Assume that the interest rate is fixed at 5% per annum. The program should display a table as shown in the sample run below. In this sample run, the user input is shaded. Assume that the user always enters valid input.

Savings	Interest earned	Amount of savings at the end of	
200	0.42	Month 1	200.83
100	0.84	Month 2	302.09
-1			

Total interest earned 1.26

The time at which you have completed Problem 3 _____

Please rate your perceived mental effort on solving this problem (circle).

1. Very low mental effort
2. Low mental effort
3. Moderate mental effort
4. High mental effort
5. Very high mental effort

³ Problem adapted from Introduction to Java Programming, 8th International Edition by Y. Daniel Liang, Pearson (2009)

APPENDIX H

Problem 4⁴

Depreciation refers to the declining value of an asset over its useful life due to wear and tear. Calculating the depreciation involves the following:

The original cost of the asset

The estimated useful life of the asset

The expected scrap value of the asset at the end of its useful life

The annual rate of depreciation

The formula for calculating the depreciation is given below:

$$\text{annual depreciation} = \text{depreciation rate} * \text{value at beginning of year}$$

Your company has bought an asset worth RM10,000. Its expected scrap value is RM1000, its estimated useful life is 5 years, and the annual depreciation rate is 40%. By the end of the first year, the asset will have declined in value to RM6000 (i.e. $10,000 - 10,000 \times 40\%$). This value becomes the new value at beginning of the second year. By the end of the second year, the asset will have declined in value to RM3600 and so on. Note that, care is needed so as to prevent the calculated value at end of the fifth year falling below the estimated scrap value. Therefore the depreciation expense at end of the fifth year is 1296 – 296. Write a program to calculate depreciation of this RM10,000 over 5 years. The program should display a table summarising the above information on depreciation as follows:

Value at beginning of year	Depreciation expense	Accumulated depreciation	Value at end of year
10000.00	4000.00	4000.00	6000.00
6000.00	2400.00	6400.00	3600.00
3600.00	1440.00	7840.00	2160.00
2160.00	864.00	8704.00	1296.00
1296.00	296.00	9000.00	1000.00

The time at which you have completed Problem 4 _____

Please rate your perceived mental effort on solving this problem (circle).

1. Very low mental effort
2. Low mental effort
3. Moderate mental effort
4. High mental effort
5. Very high mental effort

⁴ Source from Wikipedia

APPENDIX H

Program solution

Bus ticket machine

```
import java.util.Scanner;

public class BusTicket{

    public static void main(String args[])
    {
        int journey, notes;
        double amount_due, payment;
        char zone;

        amount_due = 0;
        payment = 0;

        Scanner keyboard = new Scanner(System.in);

        System.out.println("Please make a selection by choosing a letter for zone and a number for a journey\n");
        System.out.println("      1.Single  2.Return");
        System.out.println("Zone A   RM3.00   RM5.00");
        System.out.println("Zone B   RM1.00   RM3.00\n");
        System.out.println("This machine accepts RM1, RM5 and RM10 only\n");

        System.out.print("Zone: ");
        zone = keyboard.next().charAt(0);

        while (zone != 'N'){

            System.out.print("Journey: ");
            journey = keyboard.nextInt();

            if ((zone == 'A' && journey == 1) || (zone == 'B' && journey == 2)){
                amount_due += 3.00;
                System.out.printf("Amount due: RM%.2f\n", amount_due);
            }

            if (zone == 'A' && journey == 2){
                amount_due += 5.00;
                System.out.printf("Amount due: RM%.2f\n", amount_due);
            }

            if (zone == 'B' && journey == 1){
                amount_due += 1.00;
                System.out.printf("Amount due: RM%.2f\n", amount_due);
            }

            System.out.print("Zone: ");
            zone = keyboard.next().charAt(0);
        }

        while (amount_due > payment){
            System.out.print("Please insert RM notes:");
            notes = keyboard.nextInt();
        }
    }
}
```

APPENDIX H

```
switch(notes)
{
    case 10 : System.out.printf("Amount received: RM%.2f\n", payment += 10.00);
    break;
    case 5  : System.out.printf("Amount received: RM%.2f\n", payment += 5.00);
    break;
    case 1  : System.out.printf("Amount received: RM%.2f\n", payment += 1.00);
    break;
}

if(payment > amount_due){
    System.out.printf("Your change is: RM%.2f\n", payment-amount_due);
    System.out.println("Thank you");
}
else
    System.out.println("Thank you");
}
}

===== End of bus ticket machine program =====
```

APPENDIX H

Hi-Lo number guessing game

```
import java.util.Scanner;

public class SecretNumber{

    public static void main(String args[])
    {

        final int MAX_GUESSED_ALLOWED = 6;
        int secret_number, input, guess_count;
        double Y;

        guess_count=0;

        Y = Math.random();
        secret_number = (int) Math.floor(Y*100)+1;

        Scanner keyboard = new Scanner(System.in);

        do {
            System.out.print("Enter a secret number (from 1 to 100): ");
            input = keyboard.nextInt();

            while (input < 1 || input > 100) {
                System.out.println("Invalid range of number");
                System.out.println("Enter a secret number (from 1 to 100)");
                input = keyboard.nextInt();
            }

            guess_count++;

            if (input == secret_number){
                System.out.println("Congratulations");
                System.out.println("You took " + guess_count + "guesse(s).");
            }
            else
            {
                if (input < secret_number)
                {
                    System.out.println("Your guess is low.");
                }
                else
                    System.out.println("Your guess is high.");
            }

        } while (guess_count < MAX_GUESSED_ALLOWED && input != secret_number);

        if (input != secret_number)
            System.out.println("You lost. Secret number was " + secret_number);
    }
}
```

===== End of hi-lo number guessing program =====

APPENDIX H

Savings account

```
import java.util.Scanner;

public class CompoundInterest4{
    public static void main(String[] args) {

        int counter;
        double monthlyInterestRate, monthlyDeposit, currentValue, interestEarned, total_interestEarned;

        counter = 1;
        monthlyInterestRate = 0.00417;
        interestEarned = 0.0;
        total_interestEarned = 0.0;

        Scanner keyboard = new Scanner(System.in);

        System.out.println("Savings amount \tInterest earned \tAmount of savings at the end of");

        monthlyDeposit = keyboard.nextInt();
        currentValue = monthlyDeposit * (1 + monthlyInterestRate);
        interestEarned = (monthlyDeposit * (1 + monthlyInterestRate)) - monthlyDeposit;
        total_interestEarned = total_interestEarned + interestEarned;

        while(monthlyDeposit != -1) {

            System.out.printf("\t\t%.2f", interestEarned);
            System.out.printf("\t\tMonth " + counter++ + " \t\t%.2f\n", currentValue);

            monthlyDeposit = keyboard.nextInt();
            currentValue = (currentValue + monthlyDeposit) * (1 + monthlyInterestRate);
            interestEarned = (monthlyDeposit * (1 + monthlyInterestRate)) - monthlyDeposit;
            total_interestEarned = total_interestEarned + interestEarned;
        }

        System.out.printf("\nTotal interest earned %.2f",total_interestEarned);
    }
}
```

===== End of savings account program =====

APPENDIX H

Depreciation

```
public class Depreciation{

    public static void main(String args[])
    {
        int years, counter;
        double asset, salvage_value, book_value_begin, book_value_end, depreciation_rate,
        depreciation_expense, accumulated_depreciation;

        counter=1;
        depreciation_rate = 0.4;
        depreciation_expense = 0.0;
        accumulated_depreciation = 0.0;
        asset = 10000.00;
        years = 5;
        salvage_value = 1000;

        book_value_begin = asset;
        System.out.println("Book value \t\t\t\t\tBook value");
        System.out.println("Beginning of year \tDepreciation expense \tAccumulated depreciation \tEnd of year");

        while(counter <= years)
        {
            if (counter == years)
                depreciation_expense = book_value_begin - salvage_value;
            else
                depreciation_expense = depreciation_rate * book_value_begin;

            accumulated_depreciation = accumulated_depreciation + depreciation_expense;
            book_value_end = book_value_begin - depreciation_expense;

            System.out.printf("%.2f", book_value_begin);
            System.out.printf("\t\t%.2f", depreciation_expense);
            System.out.printf("\t\t\t%.2f", accumulated_depreciation);
            System.out.printf("\t\t\t%.2f", book_value_end);
            System.out.print("\n");
            book_value_begin = book_value_end;

            counter=counter+1;
        }
    }
}
```

===== End of depreciation program =====

APPENDIX I

Questionnaire

Matric no: _____ Strategy: A / B / C (circle one strategy)

Instructions: For each question, please choose one answer with respect to the strategy that you have been assigned to with LECSES.

The strategy						
		Strongly disagree	Rather disagree	Neither agree nor disagree	Somewhat agree	Strongly agree
1	I had a tendency to overestimate my understanding of the example problem and solution and the strategy helped me to resolve this tendency.	1	2	3	4	5
2	The strategy helped me to identify the relationships between underlying plan structures of the example solution and the example problem.	1	2	3	4	5
3	The strategy helped my learning on the loop topic.	1	2	3	4	5
LECSES						
		Strongly disagree	Rather disagree	Neither agree nor disagree	Somewhat agree	Strongly agree
4	My attention was drawn to various underlying plan structures of the example solution by means of the <i>masking mechanism</i> (i.e. the collapsible triangular button).	1	2	3	4	5
5	It was easy to navigate around LECSES.	1	2	3	4	5
6	LECSES' response time was acceptable.	1	2	3	4	5
7	I could finish the exercises within the time given.	1	2	3	4	5

APPENDIX I

LECSES (continue...)						
<p>For question no. 8 to 13, please choose one answer with respect to the strategy that you have been assigned to with LECSES.</p> <p>Strategy A, answer question no. 8 and 9 ONLY. Strategy B, answer question no. 10 and 11 ONLY. Strategy C, answer question no. 8 to no. 13.</p>						
		Strongly disagree	Rather disagree	Neither agree nor disagree	Somewhat agree	Strongly agree
8	The prompts and hints helped me with the explanation exercise.	1	2	3	4	5
9	Working on the explanation and reflection exercises was easy.	1	2	3	4	5
10	I was able to infer the missing parts of the program from the incomplete solution.	1	2	3	4	5
11	Working on the completion and modification exercises was easy.	1	2	3	4	5
12	Working on different exercises at different stages, that is, from explanation/reflection exercises to completion/modification exercises was reasonably easy.	1	2	3	4	5
13	Strategy A helped my learning more than strategy B.	1	2	3	4	5
The transfer test						
		Strongly disagree	Rather disagree	Neither agree nor disagree	Somewhat agree	Strongly agree
14	The loop problems were reasonably OK.	1	2	3	4	5
15	The overall time given to complete the loop problems was acceptable.	1	2	3	4	5
16	I could recall analogous worked-example problem previously studied/solved and it helped me to solve similar problem.	1	2	3	4	5

APPENDIX I

The questions
17. What did you understand the following three questions? Explain in particular what you thought the difference between them was.
<i>How much new knowledge and skill did you acquire from working on this particular problem?</i>
<i>How difficult did you find it to learn things in the recent activity?</i>
<i>Please rate your perceived mental effort on solving this problem (i.e. Very low mental effort, Low mental effort, Moderate mental effort, High mental effort, Very high mental effort)</i>

APPENDIX I

Comments and suggestions
<p>18. Please provide any further comments and suggestions about any aspect of the experiment or the system.</p>

THANK YOU

APPENDIX J

List of publications

1. Abdul-Rahman, S.S.: Towards an adaptive educational programming environment based on learning styles. Paper presented at the 9th International Conference on Intelligent Tutoring Systems (Young Researcher Track), 23-27 June 2008, Montreal, Canada.
2. Abdul-Rahman, S.S.: An adaptive example-based educational programming environment based on learning styles. Abstract presented at the 20th Annual Psychology of Programming Interest Group Conference (Doctoral Consortium), 10-12 September 2008, Lancaster, United Kingdom.
3. Abdul-Rahman, S.S., du-Boulay, B.: Schema Acquisition: Implications for the Instructional Design of Examples. Short paper presented at the 14th International Conference on Artificial Intelligence in Education (Young Researcher Track), 6-10 July 2009, Brighton, United Kingdom.
4. Abdul-Rahman, S.S., du-Boulay, B.: The Role of Worked-examples in Schema Acquisition: Implications and preliminary findings. Poster presented at the 31st Annual Meeting of the Cognitive Science Society, 29 July – 1 August 2009, Amsterdam, Netherlands.
5. Abdul-Rahman, S.S., du-Boulay, B.: Learning programming via worked-examples. Paper presented at PPIG Work-in-progress Workshop, 7-8 January 2010, Dundee, Scotland.
6. Abdul-Rahman, S.S., du-Boulay, B.: Learning programming via worked-examples: learning outcome efficiencies: Extended abstract presented at PPIG Work-in-progress Workshop, 18 – 19 April 2011, Sheffield, United Kingdom.

References

- Ainsworth, S. (2006). DeFT: A conceptual framework for considering learning with multiple representations. *Learning and Instruction, 16*, 183-198.
- Anderson, J.R. (1987). Skill acquisition: Compilation of weak-method problem solutions. *Psychological Review, 94*, 192-210.
- Anderson, J.R. (2009). Personal Communication, 29 July 2009.
- Anderson, J.R., Fincham, J.M., & Douglass, S. (1997). The role of examples and rules in the acquisition of a cognitive skill. *Journal of Experimental Psychology, 23*, 932-945.
- Allert, J. (2004). Learning style and factors contributing to success in an introductory computer science course. *Proceedings of the IEEE International Conference on Advanced Learning Technologies* (pp. 385-389): IEEE.
- Atkinson, R.K., Derry, S.J., Renkl, A., & Wortham, D. (2000). Learning from examples: instructional principles from the worked examples research. *Review of Educational Research, 70*, 181-214.
- Atkinson, R.K. & Catrambone, R. (2000). Subgoal learning and the effect of conceptual vs. computational equations on transfer. In L.R. Gleitman & A.K. Joshi (Eds.), *Proceedings of the 22th Annual Conference of the Cognitive Science Society* (pp. 591-596), Mahwah, NJ: Erlbaum.
- Atkinson, R.K. & Renkl, A. (2007). Interactive example-based learning environments: Using interactive elements to encourage effective processing of worked-examples. *Educational Psychology Review, 19*, 375-386.
- Ayres, P. (2006). Using subjective measures to detect variations of intrinsic cognitive load within problems. *Learning and Instruction, 16*, 389-400.
- Bassok, M., & Holyoak, K.J. (1989). Interdomain transfer between isomorphic topics in algebra and physics. *Journal of Experimental Psychology, 15*, 153-166.
- Bates, C. (2006). *Web-programming. Building Internet Applications* (3rd edn.). John Wiley & Sons Ltd.
- Bennedsen, J., & Caspersen, M.E. (2006). Abstraction ability as an indicator of success for learning object-oriented programming? *inroads – The SIGCSE Bulletin, 38*, 39-43.
- Berends, I.E., & van Lieshout, E.C.D.M. (2009). The effect of illustration in arithmetic problem-solving: Effects of increased cognitive load. *Learning and Instruction, 19*, 345-353.

- Bergin, S. & Reilly, R. (2005). Programming: Factors that influence success. *SIGCSE'05: Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education* (pp. 411-415), New York, USA: ACM.
- Berthold, K., Eysink, T.H.S., & Renkl, A. (2009). Assisting self-explanation prompts are more effective than open prompts when learning with multiple representations. *Instructional Science*, 37, 345-363.
- Berthold, K., & Renkl, A. (2009). Instructional aids to support a conceptual understanding of multiple representations. *Journal of Educational Psychology*, 101, 70-87.
- Berthold, K., Röder, H., Knörzer, D., Kessler, W., & Renkl, A. (2011). The double-edged effects of explanation prompts. *Computers in Human Behavior*, 27, 69-75.
- Bishop, J.M. (1998). *Java Gently: Programming Principles Explained* (2nd edn.), Addison-Wesley.
- Bishop-Clark, C. (1995). Cognitive style, personality, and computer programming. *Computers in Human Behaviour*, 11, 241-260.
- Bratfisch, O., Borg, G., & Dornic, S. (1972). Perceived item-difficulty in three tests intellectual performance capacity. Report no. 29, Institute of Applied Psychology, Stockholm.
- Brunken, R., Plass, J.L., Leutner, D. (2003). Direct measurement of cognitive load in multimedia learning. *Educational Psychology*, 38, 53-61.
- Brusilovsky, P. (2001). WebEx: Learning from examples in a programming course. In Lawrence-Fowler, W. A. & Hasebrook, J. (Eds.), *Proceedings of WebNet 2001- World Conference of the WWW and Internet* (pp.124-129), Orlando, Florida: AACE.
- Byrne, P., & Lyons, G. (2001). The effect of student attributes on success in programming. *SIGCSE Bulletin*, 33, 49-52.
- Bonar, J., & Cunningham, R. (1988). Bridge: An intelligent tutor for thinking about programming. In Self, J. (Ed.), *Artificial Intelligence and Human Learning: Intelligent Computer-Aided Instruction* (pp. 391-409). London: Chapman and Hall.
- Bowles, A., & Robertson, D. (1994). A case-based reasoning approach to supporting novice programmers. *DAI Research Paper No. 705*.
- Carbone, A., Hurst, J., Mitchell, I., & Gunstone, D. (2001). Characteristics of programming exercises that lead to poor learning tendencies: Part II. *SIGCSE Bulletin*, 33, 93-96
- Carmo, L., Gomes, A., Pereira, F., & Mendes, A. (2006). Learning styles and problem solving strategies. *Proceedings of 3rd E-Learning Conference – Computer Science Education* (pp. 7-12), Coimbra, Portugal.

Catrambone, R. (1995). Aiding subgoal learning: Effects on transfer. *Journal of Educational Psychology*, 87, 5-17.

Catrambone, R. (1998). The subgoal learning model: Creating better examples so that students can solve novel problem. *Journal of Experimental Psychology: General*, 12, 355-376.

Catrambone, R., & Holyoak, K.J. (1989). Overcoming contextual limitations on problem solving transfer. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 15, 1147-1156.

Catrambone, R., & Yuasa, M. (2006). Acquisition of procedures: The learning effects of example elaboration and active learning exercises. *Learning and Instruction*, 16, 139-153.

Caspersen, M.E., & Bennedsen, J. (2007). Instructional design of a programming course – A learning theoretic approach. In *Proceedings of the Third International Workshop on Computing Education Research* (pp. 111-122), New York, USA: ACM

Chamillard, A.T., & Karolick, D. (1999). Using learning style data in an introductory computer science course. *Proceedings of the thirtieth SIGCSE technical symposium on Computer science education* (pp. 291-295), New York, USA: ACM.

Chang, K-E., Chiao, B-C., Chen, S-W., & Hsiao, R-S. (2000). A programming learning system for beginners – A completion strategy approach. *IEEE Transaction on Education*, 43, 211-220.

Chi, M.T.H., Bassok, M., Lewis, M.W., Reimann, P., & Glaser, R. (1989). Self-explanations: How students study and use examples in learning to solve problems. *Cognitive Science: A Multidisciplinary Journal*, 13, 145 – 182.

Cierniak, G., Scheiter, K., & Gerjets, P. (2009). Explaining the split-attention effect: Is the reduction of extraneous cognitive load accompanied by an increase in germane cognitive load? *Computers in Human Behaviour*, 25, 315-324.

Cummins, D. (1992). Role of analogical reasoning in the induction of problem categories. *Journal of Experimental Psychology: Learning, Memory and Cognition*, 18, 1103-1124.

Cohen, J.W. (1988). *Statistical power analysis for the behavioural sciences* (3rd edn.). Hillsdale, NJ: Lawrence Erlbaum Associates.

Conati, S. & VanLehn, K. (2000). Toward Computer-based Support of Meta-Cognitive Skills: A Computational Framework to Coach Self-Explanation. *International Journal of Artificial Intelligence in Education*, 11, 389-415.

Corbalan, G., Kester, L., & van Merriënboer, J.J.G. (2008). Selecting learning tasks: Effects of adaptation and shared control on learning efficiency and task involvement. *Contemporary Educational Psychology*, 33, 733-756.

Corbalan, G., Kester, L., & van Merriënboer, J.J.G. (2009). Combining shared control with variability over surface features: Effects on transfer test performance and task involvement. *Computers in Human Behaviour*, 25, 290-298.

Davidovic, A., Warrant, J., & Trichina, E. (2003). Learning benefits of structural example-based adaptive tutoring systems. *IEEE Transactions on Education*, 46, 241-251.

de Jong, T. (2010). Cognitive load theory, educational research, and instructional design: some food for thought. *Instructional Science*, 38, 105-134.

de Jong, T., Ainsworth, S., Dobson, M., Van der Hulst, A., Levonen, J., & Reimann, P. (1998). Acquiring knowledge in science and mathematics: The use of multiple representations in technology-based learning environments. In M. van Someren, P. Reimann, H. Boshuizen, & T. de Jong (Eds.), *Learning with multiple representations* (pp. 9-41), Oxford: Elsevier Sciences.

de Koning, B.B., Tabbers, H.K, Rikers, R.M.J.P, & Paas, F. (2010). Attention guidance in learning from a complex animation: Seeing is understanding? *Learning and Instruction*, 20, 111-122.

DeLeeuw, K.E., & Mayer, R. (2008). A comparison of three measures of cognitive load: Evidence for separable measures of intrinsic, extraneous, and germane load. *Journal of Educational Psychology*, 100, 223-234.

Dunican, E. (2002). Making the analogy: Alternative delivery techniques for first year programming courses. In Kuljis, J., Baldwin, L., & Scoble, R. (Eds.), *14th Workshop of the Psychology of Programming Interest Group* (pp. 89-99). Brunel University.

Ehrlich, K., & Soloway, E. (1984). An empirical investigation of the tacit plan knowledge in programming. *Human Factors in Computer Systems*, 113-133, Norwood, NJ: Ablex Publishing Corp.

Felder, R.M., & Spurlin, J. (2005). Applications, Reliability and Validity of the Index of Learning Styles. *International Journal of Engineering Education*, 21, 103-112.

Felder, R.M., & Silverman, L.K. (1988). Learning and Teaching Styles In Engineering Education. *Engineering Education*, 78, 674–681.

Felder, R.M., & Soloman, B.A. (n.d.). Index of Learning Styles at <http://www.ncsu.edu/felder-public/ILSpage.html>, recently accessed on 19/8/2011.

Field, A.P. (2009). *Discovering statistics using SPSS* (3rd edn.).Sage Publications Ltd.

Garner, S. K. (2003). Learning to program using part-complete solutions. In *Proceeding of the Computer Based Learning in Science* (pp. 8-14), Nicosia, Cyprus.

Garner, S. (2007). An Exploration of How a Technology-facilitated Part-complete Solution Method Supports the Learning of Computer Programming. *Issues in Informing Science and Information Technology*, 4, 491-501.

Gerjets, P., Scheiter, K., & Catrambone, R. (2004). Designing instructional examples to reduce intrinsic cognitive load: Molar versus modular presentation of solution procedures. *Instructional Science*, 32, 33-58.

Gerjets, P., Scheiter, K., & Catrambone, R. (2006). Can learning from molar and modular worked examples be enhanced by providing instructional explanations and prompting self-explanations? *Learning and Instruction*, 16, 104-121.

Gerjets, P., Scheiter, K., & Cierniak, G. (2009). The scientific value of cognitive load theory: A research agenda based on the structuralist view of theories. *Education Psychology Review*, 21, 43-54.

Gerjets, P., Scheiter, K., & Shuh, J. (2008). Information comparisons in example-based hypermedia environments: supporting learners with processing prompts and an interactive comparison tool. *Educational Technology Research Development*, 56, 73-92.

Gilmore, D.J. (1990). Expert Programming Knowledge: A Strategic Approach. In Hoc, J.-M., Green, T.R.G., Samurcay, R., & Gilmore, D.J. (Eds.), *Psychology of Programming* (pp. 223-234). London: Academic Press Ltd.

Graf, S., Lin, T., & Kinshuk (2008). The relationship between learning styles and cognitive traits – Getting additional information for improving student modelling. *Computers in Human Behaviour*, 24, 122-137.

Graf, S., Liu, T-C., Kinshuk, Chen, N-S., & Liang, S.J.H. (2009). Learning styles and cognitive traits – their relationship and its benefits in web-based educational system. *Computers in Human Behavior*, 25, 1280-1289.

Graf, S., Viola, S.R., Kinshuk, & Leo, T. (2006). Representative characteristics of Felder-Silverman Learning Styles: An empirical model. In *proceedings on the IADIS International Conference on Cognition and Exploratory Learning in Digital Age*, Barcelona, Spain.

Gray, S., St. Clair, C., James, R., & Mead, J. (2007). Suggestions for graduated exposure to programming concepts using fading worked examples. In *Proceedings of the Third International Workshop on Computing Education Research* (pp. 99-110), New York, USA: ACM.

Große, C.S., & Renkl, A. (2006). Effects of multiple solution methods in mathematics learning, *Learning and Instruction*, 16, 122-138.

Gomez-Albarran, M. (2005). The teaching and learning of programming: A survey of supporting software tools. *The Computer Journal*, 48, 130-144.

- Honey, P. & Mumford, A. (2000). *The learning styles helper's guide*. Maidenhead: Peter Honey Publications Ltd.
- Johnson, W.L. & Soloway, E. (1985). PROUST: Knowledge-based programming understanding. *IEEE Transactions on Software Engineering*, 11, 267-275.
- Kalyuga, S., Ayres, P., Chandler, P., & Sweller, J. (2003). The expertise reversal effect. *Educational Psychologist*, 38, 23-31.
- Kalyuga, S., Chandler, P., & Sweller, J. (1998). Levels of expertise and instructional design. *Human Factors*, 40, 1-17.
- Kalyuga, S., & Renkl, A. (2010). Expertise reversal effect and its instructional implications: introduction to the special issue. *Instructional Science*, 38, 209-215.
- Keefe, J.W. (1979). Learning style: An overview. In Keefe, J.W. (Ed.), *Student learning styles: Diagnosing and prescribing programs*. NASSP.
- Kelly, A. & Pohl, I. (1996). *C by Dissection: The Essentials of C Programming* (3rd edn.), CA: Addison-Wesley.
- Kirschner, P.A. (2002). Cognitive load theory: Implications of cognitive load theory on the design of learning. *Learning and Instruction*, 12, 1-10.
- Kirschner, P.A., Ayres, P., & Chandler, P. (2011). Contemporary cognitive load theory research: The good, the bad, and the ugly. *Computers in Human Behavior*, 27, 99-105.
- Kolb, D.a. (1984). *Experiential Learning: Experience as the source of learning and development*. Englewood Cliffs, New Jersey: Prentice Hall P T R.
- Kolb, D.A. (1999). *The Kolb Learning Style Inventory, Version 3*. Boston: Hay Group.
- Lahtinen, E., Ala-Mutka, K., & Järvinen, H-M. (2005). A study of the difficulties of novice programmers. *ITiCSE'05* (pp. 14-18), Monte de Caparica, Portugal.
- Liang, Y.D. (2009). *Introduction to Java Programming* (8th International edn.), Pearson.
- Lusk, D.L., Evans, A.D., Jeffrey, T.R., Palmer, K.R., Wikstrom, C.S., & Doolittle, O.E. (2009). Multimedia learning and individual differences: Mediating the effects of working memory capacity with segmentation. *British Journal of Educational Technology*, 40, 636-651.
- Lord, F.M. (1967). A paradox in the interpretation of group comparisons. *Psychological Bulletin*, 68, 304-305.
- Lord, F.M. (1969). Statistical adjustments when comparing preexisting groups. *Psychological Bulletin*, 72, 336-337.

- Mancy, R., & Reid, N. (2004). Aspects of cognitive style and programming. In E. Dunican & T.R.G. Green (Eds.), *Proceeding 16th Workshop of the Psychology of Programming Interest Group* (pp. 1-9), Carlow, Ireland.
- Mansfield, K.C., & Antonakos, J.L. (1997). *An Introduction to Programming Using C++*. Prentice Hall.
- Mayer, R.E. (1992). *Thinking, problem solving, cognition*. New York: W.H. Freeman.
- Mayer, R.E. (2009). *Multimedia learning* (2nd edn). New York: Cambridge University Press.
- Mayer, R.E. (2010). Seeking a science of instruction. *Instructional Science*, 38, 143-145.
- Messick, S. (1984). The nature of cognitive styles: Problems and promise in educational practice. *Educational Psychologist*, 19, 59-74.
- Moreno, R. (2006). When worked examples don't work: Is cognitive load theory at an impasse? *Learning and Instruction*, 16, 170-181.
- Moreno, R. (2010). Cognitive load theory: more food for thought. *Instructional Science*, 38, 135-141.
- Moreno, R., & Valdez, A. (2005). Cognitive load and learning effects of having students organize pictures and words in multimedia environments: The role of student interactivity and feedback. *Educational Technology Research and Development*, 53, 35-45.
- Mousavi, S.Y., Low, R., & Sweller, J. (1995). Reducing cognitive load by mixing auditory and visual presentation modes. *Journal of Educational Psychology*, 87, 319-334.
- Muijs, D. (2004). *Doing quantitative research in education: With SPSS*. London: Sage Publications Ltd.
- Myers, I.B., & McCaulley, M.H. (1998). *Manual: a guide to the development and use of the Myers-Briggs Type Indicator*. Palo Alto, CA: Consulting Psychologists Press.
- Neal, L.R. (1989). A system for example-based programming. *SIGCHI Bulletin*, 20, 63-68.
- Novick, L.R. (1988). Analogical Transfer, Problem Similarity, and Expertise. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 14, 510-520.
- Oualline, S. (1997). *Practical C programming* (3rd edn.). Sebastopol, CA: O'Reilly.
- Paas, F. (1992). Training strategies for attaining transfer of problem-solving skill in statistics: A cognitive load approach. *Journal of Educational Psychology*, 84, 429-434.
- Paas, F., Renkl, A., & Sweller, J. (2003). Cognitive load theory and instructional design: Recent developments. *Educational Psychologists*, 38, 1-4.

- Paas, F., Tuovinen, J.E., Tabbers, H., Van Gerven, P.W.M. (2003). Cognitive load measurement as a means to advance cognitive load theory. *Educational Psychologist*, 38, 63-71.
- Paas, F., Tuovinen, J.E., van Merriënboer, J.J.G., & Darabi, A. A. (2005). A motivational perspective on the relation between mental effort and performance: Optimizing learner involvement in instruction. *ETR & D*, 53, 25-34.
- Paas, F., Van Gerven, P.W.M., & Wouters, P. (2007). Instructional efficiency of animation: effects of interactivity, through mental reconstruction of static key frame. *Applied Cognitive Psychology*, 21, 783-793.
- Paas, F., & van Gog, T. (2006). Optimising worked example instruction: Different ways to increase germane cognitive load. *Learning and Instruction*, 16, 87-91.
- Paas, F.G.W.C., & van Merriënboer, J.J.G. (1993). The efficiency of instructional conditions: An approach to combine mental effort and performance measures. *Human Factors*, 35, 737-743.
- Paas, F., & van Merriënboer, J.J.G. (1994a). Variability of worked examples and transfer of geometrical problem solving skills: A cognitive load approach. *Journal of Educational Psychology*, 86, 122-133.
- Paas, F., & van Merriënboer, J.J.G. (1994b). Instructional control of cognitive load in the training of complex cognitive tasks. *Educational Psychology Review*, 6, 351-371.
- Paas, F., van Merriënboer, J.J.G., & Adam, J.J. (1994). Measurement of cognitive load in instructional research. *Perception Motor Skills*, 79, 419-430.
- Pallant, J. (2007). *SPSS survival manual* (3rd edn.). Maidenhead, Berkshire: Open University Press.
- Perkins, D.N., & Martin, F. (1986). Fragile knowledge and neglected strategies in novice programmers. In Soloway, E., & Iyenger, S. (Eds.), *Empirical Studies of Programmers*. Hillsdale, NJ: Ablex.
- Pillay, N., & Jugoo, V.R. (2005). An investigation into student characteristics affecting novice programming performance. *SIGCSE Bulletin*, 37, 107-110.
- Pirolli, P. (1991). Effects of examples and their explanations in a lesson on recursion: A production system analysis. *Cognition and Instruction*, 8, 207-259.
- Pirolli, P.L., & Anderson, J.R. (1985). The Role of Learning from Examples in the Acquisition of Recursive Programming Skills. *Canadian Journal of Psychology*, 39, 240-272.
- Pollock, E., Chandler, P., & Sweller, J. (2002). Assimilating complex information. *Learning and Instruction*, 12, 61-86.

- Quilici, J.L., & Mayer, R.E. (1996). Role of examples in how students learn to categorize statistics word problems. *Journal of Educational Psychology*, 88, 144-161.
- Quilici, J.L., & Mayer, R.E. (2002). Teaching students to recognize structural similarities between statistics word problems. *Applied Cognitive Psychology*, 16, 325-342.
- Reimann, P., & Schult, T.J. (1996). Turning examples into cases: Acquiring knowledge structures for analogical problem solving. *Educational Psychologist*, 31, 123-132.
- Reisslein, J., Atkinson, R.K., Seeling, P., & Reisslein, M., (2006). Encountering the expertise reversal effect with a computer-based environment on circuit analysis. *Learning and Instruction*, 16, 92-103.
- Renkl, A. (1997). Learning from worked-out examples: A study on individual differences. *Cognitive Science*, 21, 1-29.
- Renkl, A. (1999). Learning mathematics from worked-out examples: Analysing and fostering self-explanations. *European Journal of Psychology of Education*, 14, 477-488.
- Renkl, A., & Atkinson, R.K. (2002). Learning from examples: Fostering self-explanations in computer-based learning environment. *Interactive Learning Environment*, 10, 105-119.
- Renkl, A., Atkinson, R.K., & Große, C.S. (2004). How fading worked solution steps works – A cognitive load perspective. *Instructional Science*, 32, 59-82.
- Renkl, A., Atkinson, R.K., & Maier, U.H. (2000). From studying examples to solving problems: Fading worked-out solution steps helps learning. In Gleitman, L. & Joshi, A.K. (Eds.), *Proceeding of the 22nd Annual Conference of the Cognitive Science Society* (pp 393-398), Mahwah, NJ: Lawrence Erlbaum Associates.
- Renkl, A., Atkinson, R.K., & Merrill, M.M. (2003). Transitioning from studying examples to solving problems: Effects of self-explanation prompts and fading worked-out steps. *Journal of Educational Psychology*, 95, 774-783.
- Renkl, A., Stark, R., Gruber, H., & Mandl, H. (1998). Learning from worked-out examples: The effects of example variability and elicited self-explanations. *Contemporary Educational Psychology*, 23, 90-108.
- Rist, R.S. (1989). Schema creation in programming. *Cognitive Science*, 13, 389-414.
- Ross, B.H. (1989a). Distinguishing Types of Superficial Similarities: Different Effects on the Access and Use of Earlier Problems. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 15, 456-468.
- Ross, B.H. (1989b). Reminders in learning and instruction. In S. Vosniadou & A. Ortony (Eds.), *Similarity and analogical reasoning* (pp. 438-469). Cambridge: Cambridge University Press.

Ross, B.H. (1987). This Is Like That: The Use of Earlier Problems and the Separation of Similarity Effects. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 13, 629-639.

Ross, B. & Kennedy, P. (1990). Generalizing from the use of earlier examples in problem solving. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 16, 42-55.

Salden, R.J.C.M., Aleven, V., Schwonke, R., & Renkl, A. (2010). The expertise reversal effect and worked examples in tutored problem solving. *Instructional Science*, 38, 289-307.

Salden, R.J.C.M., Paas, F., Broers, N.J., & van Merriënboer, J.J.G. (2004). Mental effort and performance as determinants for the dynamic selection of learning tasks in air traffic control training. *Instructional Science*, 32, 153-172.

Salomon, G. (1984). Television is “easy” and print is “tough”: The differential investment mental effort in learning as a function of perceptions and attributes. *Journal of Educational Psychology*, 78, 647-658.

Savitch, W.J. (2009). *Absolute Java* (4th International edn.). Pearson Education.

Scheiter, K., & Gerjets, P., & Schuh, J. (2004). The impact of example comparisons on schema acquisition: Do learners really need multiple examples? In Y. B. Kafai, W. A. Sandoval, N. Enyedy, A. S. Nixon, & F. Herrera (Eds.), *Proceedings of the 6th International Conference of the Learning Sciences* (pp. 457-464). Mahwah, NJ: Erlbaum.

Scheiter, K., & Gerjets, P. (2005). When less is sometimes more: optimal learning conditions are required for schema acquisition from multiple examples. In B. G. Bara, L. Barsalou, & M. Bucciarelli (Eds.), *Proceeding of the 27th Annual Conference of the Cognitive Science Society* (pp. 1943-1948), Mahwah, NJ: Erlbaum.

Schnotz, W. & Kürschner, C. (2007). A reconsideration of cognitive load theory. *Educational Psychology Review*, 19, 469-508.

Schworm, S., & Renkl, A. (2006). Computer-supported example-based learning: When instructional explanations reduce self-explanations. *Computers & Education*, 46, 426-445.

Seufert, T., Jänen, I., & Brünken, R. (2007). The impact of intrinsic cognitive load on the effectiveness of graphical help for coherence formation. *Computers in Human Behavior*, 23, 1055-1071.

Seufert, T., Schütze, M., & Brünken, R. (2009). Memory characteristics and modality in multimedia learning: An aptitude-treatment-interaction study. *Learning and Instruction*, 19, 8-42.

Soloway, E. (1985). From problems to programs via plans: The content and structure of knowledge for introductory LISP programming. *Journal of Educational Computing Research*, 1, 157-172.

Soloway, E. (1986). Learning to program = Learning to Construct Mechanisms and Explanations. *Communications of the ACM*, 29, 850-858.

Soloway, E., & Ehrlich, K. (1984). Empirical studies of programming knowledge. *IEEE Transactions of Software Engineering*, 10, 595-609.

Stark, R. (1999). *Lernen mit Lösungsbeispielen. Einfluß unvollständiger Lösungsbeispiele auf Beispielelaboration, Motivation und Lernerfolg* [Learning by worked-out examples. The impact of incomplete examples on example elaboration, motivation, and learning outcomes]. Bern, Switzerland: Huber.

Sweller, J. (1988). Cognitive load during problem solving: Effects on learning. *Cognitive Science*, 12, 257-285.

Sweller, J. (2010). Element interactivity, and intrinsic, extraneous, and germane cognitive load. *Educational Psychology Review*, 22, 123-138.

Sweller, J., & Cooper, G.A. (1985). The Use of Worked Examples as a Substitute for Problem Solving in Learning Algebra. *Cognition and Instruction*, 2, 59 – 89.

Sweller, J., van Merriënboer, J.J.G., & Paas, F. (1998). Cognitive architecture and instructional design. *Educational Psychology Review*, 10, 251-296.

Tarmizi, R.A., & Sweller, J. (1988). Guidance during mathematical problem solving. *Journal of Educational Psychology*, 80, 424-436.

Taiyu, Lin (n.d.). Web-OSPAN, available at <http://kinshuk.athabascau.ca/webospan/>

Thomas, L., Ratcliffe, M., Woodbury, J., & Jarman, E. (2002). Learning styles and performance in the introductory programming sequence. *SIGCSE Bulletin*, 34, 33-37.

Trafton, J.G., & Reiser, B.J. (1993). The contribution of studying examples and solving problems to skill acquisition. In Polson, M. (Ed.), *Proceedings of the 15th Annual Conference of the Cognitive Science Society* (pp. 1017-1022). Hillsdale, NJ: Erlbaum.

Turner, M. L., & Engle, R. W. (1989). Is Working Memory Capacity Task Dependent? *Journal of Memory and Language*, 28, 127–154.

Tuovinen, J.E., & Paas, F. (2004). Exploring multidimensional approaches to the efficiency of instructional condition. *Instructional Science*, 32, 133-152.

Ullman, L. (2004). *PHP for the World Wide Web* (2nd edn.). Berkeley, CA: Peachpit Press.

Valade, J. (2004). *PHP and MySQL for dummies* (2nd edn.). Indianapolis, Indiana: Wiley Publishing Inc.

Van Gerven, P.W.M., Paas, F., van Merriënboer, J.J.G., & Schmidt, H.G. (2002). Cognitive load theory and aging: Effects of worked examples on training efficiency. *Learning and Instruction*, 12, 87-105.

- Van Gerven, P.W.M., Paas, F., van Merriënboer, J.J.G., & Schmidt, H.G. (2004). Memory load and the cognitive pupillary response in aging. *Psychophysiology*, 41, 167-174.
- van Gog, T., Paas, F., & van Merriënboer, J.J.G. (2006). Effects of process-oriented worked-examples on troubleshooting transfer performance. *Learning and Instruction*, 16, 154-164.
- van Gog, T., Paas, F., & van Merriënboer, J.J.G. (2008). Effects of studying sequences of process-oriented and product-oriented worked-examples on troubleshooting transfer efficiency. *Learning and Instruction*, 18, 211-222.
- van Gog, T., & Paas, F. (2008). Instructional efficiency: Revisiting the original construct in educational research. *Educational Psychologist*, 43, 16-26.
- van Gog, T., & Rummel, N. (2010). Example-based learning: Integrating cognitive and social-cognitive research perspectives. *Educational Psychology Review*, 22, 155-174.
- VanLehn, K. (1996). Cognitive skill acquisition. Annual Review of Psychology. In Spence, J., Darly, J., & Foss, D.J. (Eds.), *Annual Reviews* (vol. 47, pp. 513-539), Palo Alto, CA.
- VanLehn, K., Jones, R.M., & Chi., M.T.H. (1992). A model of the self-explanation effect. *The Journal of the Learning Sciences*, 2, 1-59.
- VanLehn, K., & Jones, R.M. (1993). Better learners use analogical problem solving sparingly. In P.E. Utgoff (Ed.), *Proceeding of the 10th Annual Conference on Machine Learning* (pp. 338-345). San Mateo, CA: Morgan Kaufmann.
- van Merriënboer, J.J.G. (1988). Relationship between cognitive learning style and achievement in an introductory computer programming course. *Journal of Research on Computing in Education*, 29, 181-185.
- van Merriënboer, J.J.G. (1990a). Instructional strategies for teaching computer programming: Interactions with the cognitive style reflection-impulsivity. *Journal of Research on Computing in Education*, 23, 45-53.
- van Merriënboer, J.J.G. (1990b). Strategies for programming instruction in high school, Program completion vs. program generation. *Journal of Educational Computing Research*, 6, 265-287.
- van Merriënboer, J.J.G. (2009). Personal Communication, 14 November 2009.
- van Merriënboer, J.J.G., & de Croock, M.B.M. (1992). Strategies for computer-based programming instruction: Program completion vs. program generation. *Journal of Educational Computing Research*, 8, 365-394.
- van Merriënboer, J.J.G., Kirschner, P.A., & Kester, L. (2003). Taking the load off a learner's mind: Instructional design for complex learning. *Educational Psychologist*, 38, 5-13.

van Merriënboer, J.J.G., & Krammer, H.P.M. (1987). Instructional strategies and tactics for the design of introductory computer programming courses in high school. *Instructional Science*, 16, 251-285.

van Merriënboer, J.J.G., & Paas, F.G.W.C. (1990). Automation and Schema Acquisition in Learning Elementary Computer Programming: Implications for the Design of Practice. *Computers in Human Behaviour*, 6, 273-289.

van Merriënboer, J.J.G., Schuurman, J.G., de Croock, M.B.M., & Paas, F.G.W.C., (2002). Redirecting learners, attention during training: effects on cognitive load, transfer test performance and training efficiency. *Learning and Instruction*, 12, 11-37.

van Someren, M.W., Boshuizen, H.P.A., de Jong, T., & Reimann, P. (1998). Introduction. In M. van Someren, P. Reimann, H. Boshuizen, & de Jong, T. (Eds.), *Learning with multiple representations* (pp. 1-5). Oxford: Elsevier Sciences.

Vohra, D. (2008). *Ajax in Oracle JDeveloper*. Berlin Heidelberg: Springer-Verlag.

Weber, G. (1993). Analogies in an Intelligent Programming Environment for Learning LISP. In Lemut, E., du Boulay, B., Dettori, G. (Eds.), *Cognitive Models and Intelligent Environments for Learning Programming* (pp. 210-219). Berlin: Springer-Verlag.

Webb, N.M. (1989). Peer Interaction and Learning in Small Groups. *International Journal of Educational Research*, 13, 21-40.

Weber, G., & Brusilovsky, P. (2001). ELM-ART: An Adaptive Versatile System for Web-based Instruction. *International Journal of Artificial Intelligence in Education*, 12, 351-384.

Wierwille, W.W., & Eggemeier, F.L. (1993). Recommendations for mental workload measurement in a test and evaluation environment. *Human Factor*, 35, 263-281.

Wouters, P., Paas, F., & van Merriënboer, J.J.G. (2009). Observational learning from animated models: Effects of studying-practicing alternation and illusion of control on transfer. *Instructional Science*, 38, 89-104.

Wu, C.T. (2006). *An introduction to object-oriented programming with Java* (2nd edn). McGraw-Hill Higher Education.

Wylie, R., Koedinger, K.R., & Mitamura, T. (2009). Is Self-Explanation Always Better? The Effects of Adding Self-Explanation Prompts to an English Grammar Tutor. In N.A. Taatgen & H. van Rijn (Eds.). *Proceedings of the 31st Annual Conference of the Cognitive Science Society* (pp. 1300-1305), Austin, TX: Cognitive Science Society

Yousoof, M., M-Sapiyan, B., & Kamaluddin, K. (2007). Measuring cognitive load – A solution to ease learning of programming. *Proceedings of World Academy of Science, Engineering and Technology* (vol. 20, pp. 216-219).